# Serveur Apache / WEB


| | |
|---|---|
| **Auteur** | **David Parize** |
| **Date de création** | **01/02/03** |
| **Version** | **15** |
| **Date de dernière mise à jour** | **14/05/03** |


**Serveur Apache / WEB**

Ce document a été réalisé grace a la formation ***« Administration Unix / Linux / Apache »*** que j'ai suivi à **Efficom**.
Je remercie par ailleurs mes formateur ***Sébastien Dupont*** de Epita et ***Franck Desmeuze*** qui m'ont donnés cet enseignement . Je remercie les sites qui m'ont permis de réaliser ce document
www.linux-sottises.net,
www.toutestfacile.com
www.apache.org
www.php.net
www.zlib.org
www.ac-creteil.fr/reseaux/systemes/linux/
etc…
,.

Toutes suggestion peut m'être adressée à l'adresse suivante david.parize@wanadoo.fr avec comme objet de message ***Apache***

Merci

***David Parize***
david.parize@wanadoo.fr

# TABLE DES MATIERES

# ☯ PREAMBULE

Après avoir acquis ses premières lettres de noblesse avec des sites au contenu statique, le Web a depuis quelques temps amorcés un virage afin de rendre la consultation de l'information plus "adaptable" aux desiderata de tout un chacun.
Les évolutions ont dans un premier temps été apportées du côté client, tout d'abord en donnant à ***l'internaute*** le moyen d'agir via des *formulaires*, puis par des *scripts* écrits dans un langage adapté. ***Netscape*** venait de créer ***Javascript***.
***Microsoft*** proposera plus tard ***VBScript*** avec une syntaxe quasi-identique à *Visual Basic.*
Les limitations sont vite apparues (accès aux fichiers, interactions avec des bases de données) et d'autres pistes ont été étudiées. L'idée qui émergea fut ainsi de déplacer le travail d'interactivité vers le serveur **Web** : la notion de scripts exécutés côté serveur (***server-side scripting***) était née et ouvrait grand les portes du Web dynamique.
Le concept général est d'intégrer au sein d'une page statique des éléments de code écrits dans un langage spécifique, et qui vont être interprétés par un module installé sur le serveur Web.
Le résultat de cette interprétation est généralement du ***HTML***,
qui est inséré par le serveur ***Web*** en lieu et place des zones de code au moment où la page est envoyée vers le navigateur de l'internaute.
Actuellement, plusieurs grands plates-formes destinées à fournir du contenu dynamique existent sur le marché. Elles sont globalement constituées de trois composants :

- le serveur ***HTTP***,
- le module d'interprétation de scripts
- la base de données.

Voici une liste non-exhaustive des différents composants existants :

• Pour les serveurs HTTP, les plus utilisés sont ***Apache*** (auquel ce cours est dédié) et ***Internet Information Server*** de **Microsoft**,
• Pour les modules d'interprétation de scripts, ***PHP***, ***Active Server*** Pages de **Microsoft** et ***ColdFusion*** de Allaire,
• Pour les bases de données, ***MySQL*** et ***PostgreSQL***, ***SQL Server*** de **Microsoft** et **Oracle**.
La plate-forme pour les sites dynamiques la plus en vogue du moment (car elle utilise uniquement des logiciels libres) est constituée de ***Apache / PHP / MySQL***, tournant sur **Linux**.

Le présent guide a donc pour objectif de présenter une manière d'installer et configurer cette plate-forme comme support pour des sites Web dynamiques, que ce soit pour un ***Intranet*** ou un site ***Internet***.

- Nous étudierons tout d'abord l'installation des librairies ***MM***, ***Freetype***, ***GD*** et ***PDF***, dédiées respectivement à la gestion de la mémoire partagée, la génération de caractères ***True Type,*** d'images et de fichiers PDF.
- Puis nous passerons à l'installation et à la configuration de ***MySQL***.
- L'étape suivante consistera en la mise en oeuvre d'Apache, et nous finirons par l'intégration du module PHP à Apache.
- Avant toute chose, il faut vous procurer l'ensemble des sources nécessaires. Vous les trouverez dans la section Téléchargement, dans la rubrique Apache / PHP/MySQL. Vous pouvez charger l'ensemble des fichiers du tableau. Après les avoir chargées, placez-les dans un répertoire temporaire dans votre arborescence (/home/apache)
  Passer également sous le profil root afin d'avoir les droits nécessaires.

## I. INTRODUCTION

- MicroSoft — gère **l'HTML, HTM**
- ASP — Langage dynamique, utilisé dans les formulaires, lié à une base de données, dérivé du visual **BASIC**.
  L'asp fonctionne, avec **SQL Server**, **Access**,**MySql**, **Oracle**
- Apache — Fonctionne sous Linux, Unix, MicroSoft, il gère l'HTML, PHP (ASP)
  Super rapide, & gratuit
  ASP module additionnel ASPMOD, Perl
  BD : Oracle, MySql, Postgres SQL.

## II. PRE-INSTALLATION

Avant toute chose, il faut installer 4 bibliothèques nécessaires a l'utilisation de Apache.

- MM — permet la gestion de la mémoire partagée, utilisé pour les connexion IPC( communication, inter processus)
- Freetype — Donne le support des polices truetype de Windows (TTF)
- GD — Génère des images dynamiquement avec PHP
- PDFlib — Génération de fichier PDF à la volée.

# ☯ BIBLIOTHEQUES

## I. MM

### A. Introduction

**MM** est une librairie qui permet une utilisation simplifiée de la mémoire partagée (ou shared memory) utilisée lors de la communication **inter-processus** (ou IPC, Inter Processus Communication) sur les plates-formes de type Unix. La raison pour laquelle il est intéressant d'installer cette librairie sur votre système est lors de l'utilisation des sessions avec **PHP 4**.
En effet, Tobias Ratschiller, dans son article sur la gestion des sessions (l'article ici) recommande la librairie MM :
"si vous souhaitez des performances élevées, le module **mm** est une alternative très intéressante, car il permet le stockage des données de session en mémoire partagée (**NDR** : Plutôt que dans un fichier) et n'est donc pas limité par les performances du sous-système disque".
Lieu : www.engelschall.com/sw/mm mm-1.2.1.tar.gz

### B. Installation de MM

- Pour installer MM, décompactez les sources :

```
# tar xvfz mm-1.2-1.tar.gz
```

- Rendez-vous dans le répertoire qui vient d'être créé :

```
#cd mm-1.2.1
```

- Préparez la configuration, compilez la librairie et installez-la :

```
# ./configure --prefix=/usr/local
# make
# make install
```

## II. FREETYPE

### A. Introduction

La librairie **Freetype** apporte à Linux le support des polices **True Type** de Windows. Le premier champ d'application possible est leur utilisation sous **X-Window**.
L'autre possibilité, relayée par la librairie **GD**, est la génération d'images comportant un texte utilisant des polices **True Type**.

Lieu http://prdownloads.sourceforge.net/freetype/ freetype-2.1.3.tar.gz

### B. Installation de Freetype

- Pour installer cette librairie, décompactez les sources :

```
# tar xvfz freetype-2.1.3.tar.gz
```

- Rendez-vous dans le répertoire qui vient d'être créé :

```
# cd freetype-2.1.3/build/unix
```

- Préparez la configuration :

```
./install-sh unix-cc.in /home/freetype-2.1.3/
./configure
# make setup
```

- Par défaut, les composants de la librairie vont être placés dans des sous-répertoires du dossier**/usr/local**. Vous pouvez changer cela, par exemple installer dans **/usr**, en tapant à la place :

```
# make setup CFG="-prefix=/usr"
```

- Ceci fait, lancez la compilation et l'installation :

```
# make
# make install
```

- La librairie **FreeType** est une librairie dynamique, c'est-à-dire qu'elle est indépendante des applications qui l'utilisent (son code n'est pas intégré aux applications) et n'a besoin d'exister qu'une seule fois en mémoire. Une librairie dynamique est comparable aux **DLLs** de Windows. Pour la mettre à disposition du système, il faut reconstruire la liste des librairies dynamiques par la commande suivante :

```
# ldconfig
```

- Validez ensuite l'écriture des données en mémoire tampon sur le disque par la commande **sync**. Sans cela, le cache des librairies dynamiques n'est pas mis convenablement à jour. Pour vérifier la disponibilité effective de **Free Type**, tapez la commande suivante :

```
# ldconfig -v | grep freetype
```

- Le résultat devrait être

```
libfreetype.so.6 -> libfreetype.so.6.1.0.
```

## III. GD

### A. Introduction

Pour ajouter le support de la génération dynamique d'images à **PHP**, nous allons utiliser la librairie **GD**. La dernière version stable est la 1.8.4 mais celle-ci ne fonctionne pas correctement avec FreeType du fait qu'elle ne supporte par les polices True Type de type 2.
Il est donc nécessaire d'utiliser la **version 2.0.11** qui ne devrait pas poser de problèmes particuliers.
Ajoutons qu'afin de compiler cette librairie sans souci, il est nécessaire que les librairies suivantes,soient préalablement installées
**libpng**, **libjpeg-6b** et **zlib**.

#### *a) Installation de zlib-1.1.4.tar.gz*

```
verifier        rpm –qa | grep zlib
```

***lieu:*** http://www.gzip.org/zlib/ zlib-1.1.4.tar.gz

```
tar xvzf zlib-1.1.4.tar.gz
cd zlib-1.1.4.
./configure
make
make install
```

Pour ajouter le support du format TIFF dans PHP, je vous recommande si ce n'est déjà fait d'installer **libtiff**.
Lieu : www.boutell.com/gd gd-2.011.tar.gz gd-2.0.1.tar.gz
Manuel : http://www.boutell.com/gd/manual2.0.11.html

***b) Installation de jpeg-6b.tgz***

| *verifier:* ***rpm –qa \| grep libjpeg-6b*** |
|---|

*Lieu :* ftp://ftp.uu.net/graphics/jpeg/jpegsrc.v6b.tar.gz
***manuel*** http://beyond.linuxfromscratch.org/view/cvs/general/libjpeg.html

```
tar xvzf  jpeg-6b.tgz
./configure --enable-static --enable-shared --prefix=/usr &&
make &&
make install
```

***c)* Installation *de libpng-1.2.5.tar.gz***

| ***Verifier :*** ***rpm –qa \| grep libpng*** |
|---|

**Lieu:** http://www.libpng.org/pub/png/libpng.html libpng-1.2.5.tar.gz
manuel : http://www.libpng.org/pub/png/libpng-manual.doc

```
tar –xvzf libpng-1.2.5.tar.gz
cd libpng-1.2.5
cp scripts/makefile.openbsd makefile
make
make install
```

## B. Installation de GD

- Pour installer cette librairie. Décompactez les sources et rendez-vous dans le répertoire créé :

```
# tar xvfz gd-2.0.1.tar.gz
#cd gd-2.0.1
```

- GD dans sa version 2.0.11 va chercher par défaut à utiliser la librairie FreeType donc aucune configuration n'est nécessaire de ce côté. Il est par contre nécessaire d'ajuster le chemin de recherche des fichiers d'en-tête (.h).
  Pour cela,
  *éditer le fichier* ***Makefile,***
  *recherchez la ligne commençant par* ***INCLUDEDIRS***
  *modifiez*

| ***-I/usr/include/freetype2 en -I/usr/local/include/freetype2.*** ***Ligne 51*** |
|---|

- Il faut ensuite modifier l'emplacement d'installation de la librairie.
- Pour cela, rechercher la ligne ***INSTALL_LIB*** modifiez le répertoire ***/usr/lib*** vers ***/usr/local/lib.***
- Faites de même pour la ligne ***INSTALL_INCLUDE*** et changez la valeur vers ***/usr/local/include***.
- Vous pouvez maintenant lancer la compilation et l'installation :

```
# make install
```

- Comme pour FreeType, vous devez mettre à jour la base des librairies dynamiques :

```
# ldconfig
# sync
```

- Vérifiez la prise en compte de GD :

```
# ldconfig -v | grep gd
```

- Vous devriez voir apparaître

```
libgd.so.2.0.0 -> libgd.so.2.0.0
```

- Cette version de **GD2.0.11** se compile de la façon le plus suivante :

```
./configure
make
make install
ldconfig
sync
```

L'installation fonctionne très bien sur la RedHat

## IV. PDFLIB

### A. Introduction

- La librairie **PDFLib** permet à **PHP** de générer des fichiers **PDF** à la volée. Il est dès lors possible de proposer aux visiteurs du site de sauvegarder un article ou le résultat d'une requête sous la forme d'un document PDF et donc de profiter des avantages qu'offre ce format.
- Lieu : http://www.pdflib.com/pdflib/download/index.html pdflib-4.0.3 tar.gz pdflib-4.0.3-OpenBSD.tar.gz

### B. Installation de PDFLib

- Décompactez les sources et placez-vous dans le répertoire de travail :

```
# tar xvfz pdflib-4.0-3.tar.gz
#cd pdflib-4.0.3
```

- Lancez la configuration en fonction de votre plate-forme, puis la compilation :

```
# ./configure
# make
```

- Enfin, passez à l'installation :

```
# make install
```

- Comme **FreeType**, la librairie et ses composants seront placés dans des sous-répertoires de **/usr/local**.
- Il faut enfin mettre à jour la base des librairies dynamiques comme pour les deux librairies précédentes :

```
# ldconfig
# sync
```

# MySQL

## I. Introduction

Le Système de Gestion de Bases de Données Relationnelles (**SGBDR**) MySQL doit sa popularité à plusieurs facteurs.
Tout d'abord son ouverture sur le monde du logiciel libre, mais également sa simplicité et sa rapidité qui lui ont valu d'être le moteur de bases de données de choix dans les sites **Web** dynamiques.
En effet, **MySQL** offre des performances incroyables lors de la consultation de données, ce qui est un critère primordial. A contrario, il n'est que peu adapté pour des applications transactionnelles classiques (**Online** Transaction **Processing**, OLTP), le verrouillage lors d'un ajout ou d'une modification se faisant au niveau de la table toute entière
Ajoutons également qu'il ne gère pas les clés étrangères. Ces lacunes n'ont toutefois que peu d'impact dans le cadre d'un site d'information où quelques **webmestres** ajoutent ponctuellement des **news** ou des articles.
Désormais libre d'utilisation, **MySQL** évolue régulièrement et une nouvelle version dénommée **MySQL-Max** a vu le jour. Celle-ci apporte un meilleur support des transactions et élargit de ce fait le champ d'action de ce moteur aux qualités certaines.
Le principal concurrent de **MySQL** dans le monde du logiciel libre est **PostgreSQL**.
Ce **SGBDR** a une vocation de base plus généraliste que **MySQL** et se veut donc plus complet.
Ainsi, et entre autres, les clés étrangères sont supportées et la gestion des verrous est plus souple.
Par contre, il n'est pas censé offrir la même rapidité en consultation que son concurrent

Lieu : http://www.mysql.com/downloads/index.html mysql3.23.55.tar.gz
http://beyond.linuxfromscratch.org/view/cvs/content/mysql.html
Manuel : http://www.mysql.com/Downloads/Manual/manual.pdf-2002-07-26.zip

## II. Installation de MySQL

### A. Création d'un profil utilisateur

Tout d'abord, dans un souci de sécurité (surtout si votre serveur est sur Internet), il est préférable de faire tourner le moteur **MySQL** sous le profil d'un utilisateur avec des droits limités plutôt que **root**. Ainsi, dans l'hypothèse où un pirate pourrait exploiter une brèche de sécurité de ***MySQL***, son champ d'action serait limité aux droits conférés à l'utilisateur fictif.

- Tapez les commandes suivantes afin de créer un groupe **mysql** et un utilisateur **mysql**:

```
# groupadd mysql
# useradd -G mysql mysql
```

- Le choix du nom du groupe et de l'utilisateur restent à votre discrétion. Ainsi, si vous préférez **mygrp** et **myusr**, la commande à taper sera :

```
# groupadd mygrp
# useradd -g mygrp myusr
```

**Remarque** : sur une **Slackware 8.0**, même si le package **MySQL** n'a pas été installé, l'utilisateur ***mysql*** et le groupe ***mysql*** existent déjà. Cela ne pose pas de problème, mais dans le fichier **/etc/passwd**, dans la ligne correspondant à mysql, il faut remplacer ***/var/lib/mysql*** par /***usr/local/mysql***.

### B. Vérification des package sous Linux

```
rpm –qa | grep –i mysql
```

*Si vous obtenez :* *mySQL-client-3.23.47-5mdk*
*mySQL-shared-3.23.47-5mdk*
*mySQL-3.23.47-5mdk*
*php-mysql-4.1.2-1mdk*
Supprimer les paquetages :

```
rpm –e nom_package
```

- Réinstaller ces paquetages :

```
rpm –ivh    mysql-3.23.55mdk
            mysql-client
            mysql-devel
            mysqkl-shared
```

- *Mieux vaut utiliser les sources*

### C. Décompactage et compilation

- Ceci fait, décompactez les sources de MySQL et placez-vous dans le répertoire de travail :

```
# tar xvfz mysql-3.23.55.tar.gz
# cd mysql-3.23.55
```

Les possibilités de pré configuration de ***MySQL*** sont vastes. Nous allons donc vous en présenter certaines qui devraient vous permettre de déterminer lesquelles utiliser en fonction de vos besoins.

***--prefix=rép*** : **rép** sera le répertoire principal de **MySQL**, les composants seront placés dans des sous-répertoires (bin, share, etc.).
***--localstatedir=rép***: les bases de données et les journaux (***logs***) seront placés dans ce répertoire (sur un disque de données séparé par exemple).
***--without-debug*** : MySQL ne sera pas compilé pour produire des informations de débogage. Cette option est à sélectionner pour un serveur de production.
***--enable-assembler*** : lors de la compilation, des fonctions de manipulation de chaînes de caractères en assembleur seront utilisées.
***--with-client-ldflags=-all-static*** : les outils client de MySQL (***mysql, mysqladmin***) seront compilés de façon à ne pas utiliser les librairies dynamiques. Les binaires générés seront plus gros, mais les applications seront plus rapides.
***--disable-shared*** : seules les versions statiques des librairies de MySQL seront générées.

- Nous proposons de faire la pré-configuration avec la commande suivante :

```
- ./configure \
--prefix=/usr/local/mysql-3.23.55\
--localstatedir=/data/mysql \
--without-debug \
--enable-assembler \
--with-mysqld-ldflags=-all-static
```

- L'installation de ***libtermcap2-devel.rpm*** serait nécessaire

**N.B.** : les backslashes en fin de ligne permettent, en tapant sur la touche "**Entrée**", de continuer la saisie de la commande à la ligne suivante et non pas de la valider.

- Lancez ensuite la compilation puis l'installation par les commandes suivantes :

```
make
make install
```

- Pour plus de simplicité ultérieurement, nous vous suggérons de créer un lien symbolique nommé ***mysql*** vers le répertoire d'installation de **MySQL** :

```
ln -s /usr/local/mysql-3.23.55 /usr/local/mysql
```

Ainsi, toute référence à la version de MySQL est occultée, ce qui est plus souple et plus évolutif.

Les outils **MySQL** utilisant les librairies dynamiques, il est nécessaire de configurer le gestionnaire de librairies dynamiques pour qu'il connaisse l'emplacement de ces librairies.

- Pour ce faire, tapez la commande suivante :

```
echo /usr/local/mysql/lib/mysql » /etc/ld.so.conf
```

- Procédez ensuite à la mise à jour du cache des librairies :

```
ldconfig
sync
```

- La vérification du bon paramétrage se fera par la commande suivante :

```
ldconfig -v | grep mysql
```

- Le résultat devant être :

```
/usr/local/mysql/lib/mysql:
libmysqlclient.so.10 -> libmysqlclient.so.10.0.0
```

### D. Création de table et droits

L'étape suivante consiste à créer les tables système (utile uniquement si **MySQL** n'a jamais été installé), depuis le répertoire ou vous avez copié le *tar.gz* ***/home/mysql-3.23.55***

```
# scripts/mysql_Install_db
```

```
Preparing db table
Preparing host table
Preparing user table
Preparing func table
Preparing tables_priv table
Preparing columns_priv table
Installing all prepared tables
030304 16:44:19  /usr/local/mysql-3.23.55/libexec/mysqld: Shutdown Complete
```

Lors de son exécution, le script ***mysql_install_db*** vous suggère vivement de saisir un mot de passe pour l'utilisateur ***root*** de **MySQL**. Cet utilisateur dispose de tous les privilèges sur les tables systèmes donc ce conseil est bon à suivre.

Lancez les commandes suivantes pour donner les droits sur répertoires de MySQL aux utilisateurs fictifs créés précédemment :

```
# chown -R root /usr/local/mysql/
# chown -R mysql /data/mysql                    répertoire par défaut /usr/local/mysql/var
# chgrp -R mysql /usr/local/mysql/
# chgrp -R mysql /data/mysql
```

Le ***"/"*** à la fin des commandes portant sur le répertoire ***/usr/local/mysql*** est important car il permet que le traitement s'opère sur le répertoire pointé par le lien **mysql** et non le lien lui-même.

### E. Configuration

- Les exécutables sont installé par défaut dans /usr/local/bin
- *Le daemon mysql* */usr/local/libexec*
- *Les bibliothèque dans* */usr/local/lib/mysql*
- *Les base sont installées dans* */usr/local/var*
-

Si vous disposez de 512 Mo à 1 Go de mémoire, utilisez le fichier ***my-large.cnf***. Si vous disposez de plus d'1 Go de mémoire, utilisez le fichier ***my-huge.cnf.***

- Copiez ensuite le fichier de configuration à sa place :

```
# cp support-files/my-medium.cnf  /etc/my.cnf
```

- Editez le fichier***/etc/my.cnf*** et ajoutez la ligne suivante dans la section

```
[mysqld] : user= mysql
```

Ceci permet de spécifier au démon **MySQL** de tourner avec les droits de l'utilisateur ***mysql*** sans avoir à le spécifier sur la ligne de commande.

Une astuce pour gagner en performance : utilisez la commande Unix ***strip*** qui retire d'un binaire les symboles utilisés lors de l'édition de lien. Le plus utile est de l'exécuter sur ***mysqld***, le démon MySQL :

```
strip /usr/local/mysql/libexec/mysqld
```

### F. Démarrage

- Vous pouvez enfin démarrer le démon MySQL par la commande suivante :

```
# /usr/local/mysql/bin/safe_mysqld &
```

- Si tout fonctionne bien, MySQL devrait démarrer et vous afficher un message indiquant le répertoire utilisé pour stocker les données *(/data/mysql* dans notre cas). Pour vérifier si le démon ***mysqld*** tourne, utiliser la commande suivante :

```
# ps | grep mysql
```

Le résultat devrait ressembler à ceci : ***13653 pts/0 00:00:00 safe_mysqld***

Pour que le démon **MySQL** démarre et s'arrête automatiquement, vous pouvez utiliser le fichier **mysql.server.**

#### *a) Pour une Slackware*

- Copiez-le dans le répertoire contenant les scripts de démarrage :

```
# cp support-files/mysql.server /etc/rc.d
```

- Appliquez des droits en exécution :

```
# chmod 755 /etc/rc.d/mysql.server
```

- Editez le fichier ***rc.local*** et ajoutez les lignes suivantes :

```
# Démarrage MySQL.
if [ -x /etc/rc.d/mysql.server ]; then
. /etc/rc.d/mysql.server start
fi
```

#### *b) Pour Linux*

- Copier le fichier ***mysql.sever.sh*** dans ***/etc/rc.d/init.d*** et on le nommera ***mysql***.

```
cp mysql-3.23.55/support-files/mysql.server /etc/rc.d/init.d/mysql
```

- Donner les droits d'execution à ce fichier

```
chmods 755 /etc/rc.d/init.d/mysql
```

- Lancement automatique à l'état de marche 3—5

```
chkconfig --level 345 mysql on
```

- Et un arrêt à l'état de marche 0- 1-2-6

```
chkconfig --level 0126 mysql off
```

### G. Mot de passe root et test

- La commande **mysql** ouvre une session cliente en ligne de commande, appelé *mode console*.Passer les commandes mysql suivantes comme root et comme utilisateur quelconque, et comparer.

```
$ mysql
mysql> show databases;
mysql> use mysql;
mysql> show tables;
mysql> describe user;
mysql> select  host, user, password from user;
```

- Vous pouvez le faire par les commandes suivantes après avoir lancer le daemon *(voir Démarrage*):

```
 /usr/local/mysql/bin/mysqladmin -u root password new_password
/usr/local/mysqladmin –u root –h nom_machine –p mysql
Enter password :
Welcome to the MySQL monitor
mysql>  select  host, user, password from user;
```

On observe alors le mot de passe de root crypté !

- Vous pouvez également lancer quelques commandes de test :

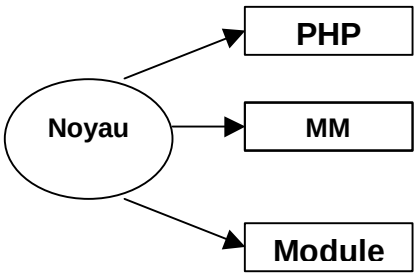
```
cd /sql-bench ; run-all-tests
/usr/local/mysql/bin/mysqladmin version
/usr/local/mysql/bin/mysqladmin variables
/usr/local/mysql/bin/mysqlshow
 /usr/local/mysql/bin/mysqlshow mysql
 /usr/local/mysql/bin/mysql -e "sélect host,db,userfrom db" mysql
```

# APACHE

## I. INTRODUCTION

Le serveur HTTP Apache jouit d'une notoriété indiscutable dans le monde d'Internet. D'après la société **Netcraft**, qui étudie chaque mois les "parts de marché" des différents type de serveurs Web, **Apache** domine largement ses concurrents avec une utilisation sur plus de 60% des serveurs **HTTP** publics dans le monde.
Ce succès est non seulement dû à sa ***gratuité***, mais également à sa ***robustesse*** et à son ***extensibilité***. Apache est également une plate-forme de choix pour un **intranet**, et est très répandu dans les universités et les écoles. Actuellement en version 1.3.27, Apache connaît des développements réguliers; Les plus grandes évolutions devraient venir de la version 2.0 (actuellement en bêta) qui apporte entre autres un support très poussé des processus légers (threads) pour une meilleure efficacité d'ensemble.

Lieu : http://www.apache.org/dist/httpd/ apache_1.3.27.tar.gz
Manuel : http://www.openbsd.org/docum.html




## II. INSTALLATION DE APACHE

### A. Décompactage et compilation

Pour installer Apache, décompactez tout d'abord les sources et placez-vous dans le répertoire nouvellement créé :

```
# tar xvfz apache_1-3.27.tar.gz
# cd apache_1.3.27
```

Tout comme **MySQL**, **Apache** accepte un nombre considérable de paramètres lors de sa pré-configuration.
Voici un exemple de commande, vous trouverez le détail de ses paramètres ci-après :

```
# ./configure \
--prefix=/usr/local/apache \          Répertoire finale d'Apache
--htdocsdir=/data/web/www \           Répertoire racine des fichiers consultables
--cgidir=/data/web/cgi-bin \          Répertoire racine des fichiers CGI
--enable-module=rewrite \             Réécriture de l'URL
--enable-shared=max                   Mettre un maximum de modules dynamique
```

**N.B.** : les *backslashes* en fin de ligne permettent, en tapant sur la touche "Entrée", de continuer la saisie de la commande à la ligne suivante et non pas de la valider.

Quelques explications sur les options :

- ***--prefix=/usr/local/apache-1.3.27*** indique que l'ensemble des composants d'Apache (binaires, modules, fichiers de configuration, etc.) seront placés dans le répertoire ***/usr/local/apache-1.3.27*** ou ses sous-répertoires (***bin***, ***conf***, ***libexec***, etc.). Si vous souhaitez installer **Apache** dans un autre endroit de votre arborescence, il suffit de jouer sur ce paramètre.

- ***--htdocsdir*** et ***-cgidir*** permettent de spécifier des répertoires où les pages HTML et les programmes ***CGI*** seront stockés.

- Pour info, si vous souhaitez placer les fichiers de configuration dans un endroit précis, vous pouvez spécifier le paramètre ***--sysconfdir=répertoire***, comme par exemple ***--sysconfdir=/etc/apache-conf*** pour placer les fichiers de configuration dans le répertoire /***etc/apache-conf***.

- ***--enable-module=rewrite*** spécifie d'ajouter le module permettant la réécriture automatique d'URL (il n'est pas disponible par défaut). Les modules de bases sont normalement suffisants et ne nécessitent pas de librairies complémentaires.

- Vous pouvez choisir l'option -***enable-module=all***, ainsi l'ensemble des modules complémentaires (*authentification*, *redirection*, etc.) vont être compilés et seront actifs, c'est-à-dire prêt à être chargés au besoin par ***Apache*** pour utiliser leurs fonctions.

Pour avoir l'ensemble des fonctionnalités, on préférera en général ***l'option --enable-module=most*** qui ajoute et active tous les modules sauf les suivants :

  - ***auth_db*** (car tous les systèmes n'ont pas la librairie db),
  - ***mmap_static*** (toutes les systèmes n'ont pas accès à la fonction mmap() - de plus ce module est considéré comme expérimental),
  - ***so*** (pas de gestion dynamique des modules - voir un peu plus bas),
  - ***example*** (utile uniquement aux développeurs de modules), ***auth_digest*** (car ce module peut entraîner des conflits avec le module ***digest*** - de plus il est considéré expérimental),
  - ***log_agent*** et ***log_referer*** (qui sont obsolètes, leurs fonctionnalités ont été intégrées dans le module ***log_config***)
  -

- ***--disable-module*** permet de spécifier certains modules optionnels fournis avec les sources d'Apache à ne pas compiler (et donc indisponibles).

- Par exemple, si vous spécifiez --enable-module=all, il vaut mieux malgré tout désactiver certains modules. Ainsi, ajouter les options

```
--disable-module=mmap_static \ --disable-module=auth_digest
--disable-module=example \
```

spécifie de ne pas utiliser les modules ***mod_mmap__static, mod_auth_digest*** et ***mod_example***.

La dernière option est certainement la plus intéressante car elle indique au configurateur que l'on souhaite utiliser les modules dynamiques.
En effet, tout comme le noyau de **Linux** est capable d'utiliser des modules qui ne sont chargés que si besoin est, **Apache** est capable d'utiliser un mécanisme similaire. Les avantages sont doubles.
Tout d'abord, **Apache** ne charge en mémoire que les modules dont il a besoin à un instant précis, et cela libère donc la mémoire centrale du serveur pour d'autres tâches.
Ensuite, on dispose d'une plus grande souplesse. Installer un nouveau module ne nécessite que la compilation du dit module et ne nécessite donc plus les sources d'Apache.
De plus, une mise à jour de version de **PHP** est bien plus aisée puisqu'il suffit de recompiler la nouvelle version sous forme de module et ne nécessite donc pas d'interruption du ***serveur HTTP***.
La légère contrepartie par rapport à une intégration des composants au sein **d'Apache** est une légère augmentation de la charge du serveur dû à la nécessite de gérer les chargements et déchargements des modules en mémoire.
Si vous souhaitez disposer d'un bon compromis entre performance et maintenance aisée, vous pouvez intégrer les modules fournis avec **Apache** (donc dépendants de la version d'Apache) dans l'exécutable.
Pour cela, il suffit de ne pas spécifier l'option ***--enable-shared=max***.

Par contre, les modules tiers (et donc indépendants de la version d'Apache) tels que ***PHP***, ***mod_gzip***, etc. peuvent être gérés comme modules dynamiques via ***apxs***.

Vous pouvez maintenant lancer la compilation et l'installation :

```
# make
# make install
```

```
+--------------------------------------------------------+
| You now have successfully built and installed the      |
| Apache 1.3 HTTP server. To verify that Apache actually |
| works correctly you now should first check the         |
| (initially created or preserved) configuration files   |
|                                                        |
|   /usr/local/apache/conf/httpd.conf                    |
|                                                        |
| and then you should be able to immediately fire up     |
| Apache the first time by running:                      |
|                                                        |
|   /usr/local/apache/bin/apachectl start                |
|                                                        |
| Thanks for using Apache.       The Apache Group        |
|                                http://www.apache.org/  |
+--------------------------------------------------------+
```

Tout comme nous l'avons fait pour **MySQL**, nous vons conseillons de **créer un lien** symbolique apache vers le répertoire d'installation (apache-1.3.27) le cas échéant :

```
# ln -s /usr/local/apache-1.3.27 /usr/local/apache
```

### B. Démarrage et tests

Vous pouvez maintenant démarrer Apache :

```
# /usr/local/apache/bin/apachectl start
```

- Pour vérifier que l'installation est correcte, ouvrez votre **navigateur** et tapez simplement **l'adresse IP** ou le nom (si vous avez un ***DNS***) de la machine sur laquelle vous venez d'installer **Apache**. La page de démarrage devrait apparaître
  Utiliser ***lynx*** sous openBSD :

```
lynx 127.0.0.1
```

- Pour **qu'Apache** démarre automatiquement, ajoutez les lignes suivantes au fichier ***/etc/rc.local*** (valable sur ***Linux Slackware***) :

```
# Démarrage Apache
if [ -x /usr/local/apache/bin/apachectl ]; then
. /usr/local/apache/bin/apachectl start
fi
```

## III. REPERTOIRE D'APACHE

Dans /usr/local/apache

| | |
|---|---|
| */bin* | Contient tous les exécutables |
| */cgi-bin* | Contient tous les scripts **CGI** |
| */conf* | Contient tous les fichiers de configuration |
| */htdocs* | Contient toutes les pages **Web** *(htm, html, php)* |

- Dans le répertoire */udsr/local/apache/htdocs*
  tapez *:* ***mv index.html.fr index.html***

| | |
|---|---|
| */icons* | contient des icônes qui servent notamment pour identifier les types de fichiers |
| */includes* | Contient tous les includes **d'Apache** |
| */libexec* | Contient les modules |
| */logs* | Contient les fichiers journaux |

*/man* Aide d'apache
*/proxy* Apache peut être utilisé comme ***Proxy***, c'est le répertoire cache pour cette fonction.

Le répertoire de log contient essentiellemnt deux fichiers :
***access.log*** listant les accès au serveur
***error_log*** listant les erreurs en tout genre

Le répertoire de module ***libexec*** contient tout les modules utilisables par ***Apache***.

**N.B** :un module est une extension logicielle à Apache, lui permettant par exemple d'interpréter le PHP4 (module **libphp4.so),** ou alors de créer des hôtes virtuels (module **mod_vhost_alias.so).**

## IV. MODULES APACHE

Ci-dessous, une liste des modules faisant partie de la distribution Apache. Voir aussi la liste des modules par ordre alphabétique de toutes les directives d'Apache (http://httpd.apache.org/docs/mod/directives.html). Pour les modules d'Apache qui ne font pas partie de la distribution, vous pouvez consulter http://modules.apache.org

### A. Base

Base Fonctionnalités de base d'Apache

### B. Création de l'environnement

mod_env Apache 1.1 et sup Passage d'environnement aux scripts CGI
mod_setenvif Apache 1.3 et sup Définition de variables d'environnement en fonction des informations client
mod_unique_id Apache 1.3 et sup Génération d'identifiants uniques de requête

### C. Gestion du type de contenu

mod_mime Détermination du type des documents en fonction de l'extension du fichier
mod_mime_magic Détermination du type des documents en fonction de "nombres magiques"
mod_negotiation Négotiation de contenu

### D. Transformation d'URL

mod_alias Association de différentes parties du système de fichier de l'hôte dans l'arborescence des documents, et redirection des URL.
mod_rewrite Apache 1.2 et supérieur Association des URI à des fichiers en utilisant des expressions régulières
mod_userdir Répertoires personnels d'utilisateurs
mod_speling Apache 1.3 et + Correction automatique d'erreurs de frappe mineures dans les URL
mod_vhost_alias Apache 1.3.7 et + Support d'hôtes virtuels dynamiquement configurables

### E. Gestion des répertoires

mod_dir Gestion de base des répertoires
mod_autoindex Création automatique des listes des répertoires

### F. Contrôle d'accès

mod_access Contrôle d'accès basé sur le nom du client ou son adresse IP
mod_auth Authentification des utilisateurs à partir d'un fichier texte
mod_auth_dbm

## V. CONFIGURATION

L'étape suivante va consister à modifier le fichier de configuration **d'Apache** afin de l'adapter à nos besoins. Celui-ci se trouve dans le sous-répertoire ***/usr/local/apache/conf*** et se nomme ***httpd.conf***. Plutôt que d'en détailler chaque ligne, nous allons vous en présenter les options (ou **directives**) importantes, ainsi que quelques propositions de modification. Le fichier ***httpd.conf*** comporte trois grandes sections que nous allons maintenant décrire.

- Pour chercher l'emplacement de clauses particulières, il est commode d'interroger ce fichier avec l'utilitaire *grep*. Par exemple pour chercher les numéros de lignes où se trouve le mot-clé *UserDir*, puis le mot *alias*, sans tenir compte de la casse :

```
grep -n "UserDir" /etc/httpd/conf/httpd.conf
grep -ni "alias" /etc/httpd/conf/httpd.conf
```

### a) Global Environment

Cette section comprend des paramètres qui décrivent le fonctionnement global **d'Apache** comme par exemple le nombre de requêtes concurrentes qu'il peut accepter. Voici les plus importants :

| | |
|---|---|
| ***ServerType standalone*** | ligne 52 |

Cette option décrit le mode d'exécution d'Apache. Les valeurs possibles sont ***inetd*** ou ***standalone*** (autonome), ce dernier étant choisi par défaut.
Rappelons que le démon ***inetd*** (appelé également le "*super-serveur Internet*") est en quelque sorte un superviseur. Il attend des connexions sur un certain nombre de ports (**21** pour ***FTP***, **23** pour ***Telnet***, etc.) et lors d'une demande de connexion sur un port particulier, il démarre une application (ou démon) spécifique apte à fournir les services qu'attend la machine appelante.
Cette application terminée, ***inetd*** continue l'écoute sur le port. L'intérêt ***d'inetd*** est donc de limiter la charge du système en ne démarrant un démon qu'à la demande.
En mode autonome, le démon **Apache** est en permanence présent en mémoire, qu'il y ait ou non des requêtes. Ce mode est préféré car plus performant :
lors d'une requête, Apache répond instantanément. A l'opposé, en mode **inetd**, un temps de latence inhérent à l'activation du démon est constaté et pénalise lourdement la performance d'un serveur très sollicité. Nous vous conseillons de conserver le paramètre ***standalone***.

| | | |
|---|---|---|
| ***ServerRoot "/usr/local/apache-1.3.27"*** | ou “/usr/local/apache” | ligne 63 |

Cette option spécifie le répertoire principal d'Apache. Les composants sont placés dans des sous-répertoires :

- ***bin*** pour les programmes,
- ***conf*** pour les fichiers de configuration,
- ***logs*** pour les journaux d'accès ou d'erreur,
- ***libexec*** pour les modules dynamiques,
- ***man*** pour les pages de manuel,

Cette option prend comme valeur celle spécifiée par le paramètre "***--prefix***" lors de la préconfiguration.

### b) Les modules dynamiques

Ces paramètres permettent l'utilisation des modules dynamiques (ou **Dynamic Shared Objects**, **DSO**). L'ordre dans lequel ils sont spécifiés est très important et il ne faut donc pas le modifier.

| | |
|---|---|
| ***LoadModule nommodule_modute libexec/nommodule.so*** | ligne 205 |
| ***ClearModuleList*** | ligne 227 |
| ***AddModule nommodule.c*** | |

Si vous souhaitez ajouter un nouveau module à Apache et que vous le compilez pour qu'il soit dynamique, il faudra ajouter les paramètres **"LoadModule**" et "**AddModule**" correspondants.
Prenons par exemple un module **mod_toto** : vous devrez ajouter la ligne juste après la dernière ligne "***AddModule ...***".

| | | |
|---|---|---|
| ***LoadModule mod_toto_module libexec/mod_toto.so*** | | ligne 271 |
| ***ClearModuleList*** | | |
| ***AddModule mod_toto.c"*** | juste après la dernière ligne "AddModule ...". | |

Actuellement, la plupart des modules tiers sont fournis sous la forme d'un fichier source (*mod_toto.c* par exemple). Pour l'installer simplement, utilisez la commande ***apxs*** disponible dans le répertoire ***bin*** de **Apache** :

| **# /usr/local/apache/bin/apxs -c -i -a mod_toto.c** |
|---|

Le module va été compilé ***(-c),*** installé dans le bon répertoire ***(-i)*** et activé ***(-a***). Cette dernière option correspond à l'ajout des lignes "***LoadModule***" et "***AddModule***" adéquates dans le fichier ***httpd.conf.***

| ***httpd –l*** | Permet de lister les modules |
|---|---|

### *c) 'Main' server configuration*

Cette section décrit l'ensemble des paramètres utilisés par le serveur par défaut, c'est-à-dire celui qui répond aux requêtes qui ne sont pas adressées à un *hôte virtuel*. Si vous souhaitez héberger plusieurs sites distincts sur un seul serveur, vous aurez besoin des hôtes virtuels.
Nous les détaillerons lors de la présentation de la **cinquième section**, qui leur est consacrée.
Si vous n'hébergez qu'un seul site, seule cette rubrique est importante. Nous ajouterons également que l'ensemble des options de configuration du serveur principal peut être repris dans la configuration d'un hôte virtuel.

| ***Port 80*** | ligne 278 |
|---|---|

Cette option spécifie le port sur lequel le serveur Apache attend des requêtes. 80 est la valeur par défaut et correspond à la valeur attribuée au service ***www*** par les instances de normalisation *(RFC 1700)*. A noter que comme pour tout port inférieur à ***1024***, un programme devant utiliser le port ***80*** doit être lancé avec les droits de root. Si vous n'avez pas les droits d'administration sur la machine, utilisez une valeur supérieure, par exemple ***8080***.

| ***User nobody***<br>***Group nobody*** | ligne 292 |
|---|---|

Les directives ***User*** et ***Group*** permettent de spécifier un utilisateur et un groupe fictif qui détermineront les droits d'accès du serveur ***Apache***, le but étant de limiter les risques en cas d'intrusion par un pirate. Nous sommes donc ici dans le même cas de figure que celui présenté précédemment avec ***MySQL***. Cependant, et comme nous l'avons évoqué ci-dessus, il n'est pas possible d'utiliser le ***port 80*** si ***Apache*** ne tourne pas avec les droits ***root***. La subtilité réside dans le mécanisme de fonctionnement ***client/serveur d'Apache***.
Un processus père, tournant avec les droits ***root***, attend des connexions sur le ***port 80.*** Lorsqu'une demande de connexion arrive, le *processus père* crée un *processus fils* (tournant avec les droits ***nobody***) qui va être chargé de satisfaire les demandes du client venant de se connecter. C'est donc un processus ayant des droits restreints qui lit les fichiers (***HTML***, ***PHP***, etc.) et les envoie au navigateur de l'internaute.
Une règle doit donc être observée : *les fichiers constituant votre site doivent être accessibles en* ***lecture*** *à l'utilisateur et/ou au groupe* ***nobody****.*
Assurez-vous que les nouveaux fichiers que vous copiez sont

| | |
|---|---|
| ***chmod a+r fichier.html*** | en lecture pour tout le monde |
| ***chown nobody fichier.html*** | appartiennent à l'utilisateur nobody |
| ***chgrp nobody fichier.html*** | ou au groupe nobody |

| ***ServerAdmin david@localdomain*** | ligne 300 |
|---|---|

La directive ***ServerAdmin*** permet simplement de définir une adresse émail qui sera affichée sur les pages générées par Apache (pages d'erreur, présentation de répertoire, etc.)

| ***ServerName openbsd.localdomain*** | ligne 318 |
|---|---|

Cette directive permet de forcer un nom d'hôte qui sera renvoyé au navigateur client,par exemple ***www*** plutôt que le nom d'hôte réel.
Attention toutefois, vous ne pouvez pas spécifier n'importe quel nom, celui que vous avez choisi devra être défini dans le ***DNS*** local. Si ce n'est pas le cas, il faudra y accéder par son *adresse IP*.
Si vous laissez la ligne commentée (avec la # devant), le serveur répondra avec le nom que vous avez tapé dans votre navigateur.
Dans un premier temps, laissez cette ligne commentée.

| ***DocumentRoot "/data/www/html"*** | ligne 325 |
|---|---|

Cette option permet de spécifier dans quel répertoire de base par défaut sont stockées les pages présentes à la racine du site (par exemple *http://nomserveur/test.html*). Utiliser lorsque l'URL ne comporte pas de chemin de répertoire.Ne changez pas cette option pour l'instant.

***PidFile /var/run/httpd.pid***

C'est le fichier ou le serveur en execution stocke son prmier numéro de processus (PID).

### *d) Controle des accès à un répertoire (ligne 403)*

| ***<Directory"/data/www/html">...</Directory>*** | ligne 350 |
|---|---|

Ces directives délimitent une liste d'options qui va s'appliquer à un répertoire spécifique.
La directive **"*Options*"** va déterminer les comportements généraux du répertoire.
La directive ***"AllowOverride"*** permet de spécifier si certaines options définies par la directive ***Options*** vont pouvoir être surchargées par l'utilisation d'un fichier d'accès (voir *AccessFileName* ci-dessous).
Les directives ***"Allow from"*** et ***"Order"*** déterminent qui a accès aux fichiers contenus dans ce répertoire, et dans quel ordre les autorisations sont gérées : d'abord les exclusions ou d'abord les autorisations.

- Chaque répertoire auquel Apache accéde peut etre configuré particulièrement (ceci s'applique aussi à ses sous-répertoires) Le paramétrage de *rep* se précise dans un conteneur noté

```
<Directory rep>
</Directory>
```

---------- Global Access Configuration ----------
Voici un exemple de paramétrage des permissions d'accès. Il est préférable d'etre restrictif et d'étendre les permissions à des répertoires spéciaux.

```
<Directory />
Options None
AllowOverride None
</Directory>
```

Pour la racine du serveur:

```
<Directory /home/httpd/html>
```

Options possibles : "None", "All", ou plusieurs combinaisons de: "Indexes", "Includes", "FollowSymLinks", "ExecCGI", ou "MultiViews".

```
Options Indexes Includes FollowSymLinks
```

***AllowOverride*** = All pour donner la priorité aux fichiers .htaccess

```
AllowOverride All
order allow,deny
```

allow from = all pour permettre à tout le monde d'accéder aux documents

```
allow from all
</Directory>
```

# pour le répertoire contenant les scripts:

```
<Directory /home/httpd/cgi-bin>
AllowOverride None
Options ExecCGI
</Directory>
```

- Les paramétres d'**Options** permet de controler l'action d'Apache sur les répertoires

| Option | signification |
|---|---|
| All \| None | toutes \| aucune option(s) peermise(s) |
| ExecCGI | exécution de scripts autorisée |
| FollowSymLinks | le serveur suivra les liens symboliques rencontrés dans le répertoire |

| | |
|---|---|
| Includes | permet l'utilisation de SSI |
| IncludesNOEXEC | permet l'utilisation de SSI sauf les directives #exec et #include |
| Indexes | autorise l'affichage d'un répertoire (si un fichier par défaut n'y est pas trouvé) |

- **AllowOverride** précise la manière avec laquelle des directives contenues dans un fichier .htacces seront prises en compte, si ces directives supplantent ou non celles qui sont dans le présent conteneur.
  Par défaut par sécurité, onn choisit la valeur NONE, et on positionne les valeurs ALL ou AuthConfig qu'au cas par cas, pour forcer une authentification à un site privé.

- **Restriction des accès** Pour un répertoire donné, dans son conteneur <Directory>, on peut préciser les hotes dont les requetes seront traitées, et ceux dont les requetes seront rejetés. On précise d'abord une règle générale avec la directive **order allow, deny** ou l'inverse, qui précise comment traiter une machine qui ne figure dans aucune liste allow ou deny explicitement.. Suivent les listes explicites après les clauses **allow from** et **deny from**
  - **order allow, deny** : autorise les hotes de la liste allow, rejette tous les autres
  - **order allow, deny** : rejette les hotes de la liste deny, autorise tous les autres

***e) Personnalisation***

⇒ **Les pages WEB personnelles**

| ***<IfModule mod_userdir.c>...</IfModule>*** | ligne 371 |
|---|---|

Ces directives délimitent une liste d'options à n'appliquer que si le nom de module spécifié (***mod_userdir*** dans ce cas) a été activé au niveau de la première section par les directives

| ***LoadModule et AddModule*** | ligne 379 |
|---|---|

| ***UserDir public_html*** |
|---|

Dans le cas où le module ***mod_userdir*** est chargé, cette option permet de spécifier un nom de répertoire qui, s'il existe dans le répertoire de base ***(home directory***) d'un utilisateur local du serveur **Apache**, va lui permettre d'y stocker des pages.Pour y accéder, il suffira de saisir comme URL http://nomserveur/~nomutilisateur/.
**Exemple** :
Utilisateur : toto
Signifie que l'utilisateur ***toto*** peut publier ses pages WEB personnelles dans un sous-répertoire de son répertoire nommé ***« public_html »***, c'est à dire dans ***/home/toto/public_html***
Sa page d'accueil sera accessible par l'URL : ***http://serveur/~toto***

Attention, le serveur **Apache** ayant les droits de l'utilisateur ***nobody***, le répertoire de base ainsi que le répertoire pour les pages des utilisateurs devront lui être accessibles. Pour ce faire, donnez leur les droits en lecture et en parcours :

```
# chmod 755 -nomutilisateur
# chmod 755 ~/public_html
ou
chmod 771 ~/public_html
```

Puis donnez les droits en lecture aux fichiers contenus dans le répertoire **public_html**

```
# chmod -R 755 ~/public_html/*
ou
# chmod 644 ~/public_html/*
```

Si vous ne souhaitez pas utiliser cette fonctionnalité, commentez les lignes ***LoadModule*** et ***AddModule*** correspondantes dans la première section du fichier ***httpd.conf.***

- Comment permettre aux utilisateurs de publier leurs "pages persos" sur le WEB de l'établissement ?

Tout naturellement, ces documents devront être placés dans leur répertoire personnel dont ils sont propriétaires et ont le plein accès, tout en en permettant l'accès et la lecture à tous.

La page d'accueil doit être présente dans un sous-répertoire du répertoire perso. Par défaut le nom de ce sous-répertoire est fixé à **public_html**
Ce sous-répertoire des pages personnelles a un nom spécifié par le paramètre **UserDir**dans le fichier httpd.conf . Donc par défaut, on y trouve :

**UserDir public_html**

Plus concrétement, si la page d'accueil s'appelle *index.html*, pour l'utilisateur *toto* , sur le serveur *www.cybergate_paradize.fr*, son ouverture sur une station du réseau est obtenue en spécifiant l'URL :
**http:// www.cybergate_paradize.fr,/~toto/index.html**, ou plus simplement : **cybergate_paradize /~toto**
On peut n'autoriser l'accès qu'à certains utilisateurs avec les lignes supplémentaires :

***UserDir disable***
***UserDir enable jean toto ....***

Attention aux droits d'accès ! L'accès de tous à ces sites persos exige que le droit de parcours x soit accordé à tous les rép. des utilisateurs et à **public_html**(droits 771), et que le droit de lecture r soit bien sûr accordé aux fichiers des pages pour tous (droits 644)

En utilisant **UserDir /home/*/public_html**, où * est remplacé par un nom d'utilisateur, celui-ci n'aplus besoin de posséder un compte sur le système.

***DirectoryIndex index.html*** ligne 404

Si le module ***mod_dir*** est chargé (*voir* ***IfModule*** *ci-dessus*), cette directive permet de spécifier des noms de fichiers qu'***Apache*** va considérer comme noms de page par défaut.

Ceci permet de taper une ***URL*** sans spécifier un nom de fichier supplémentaire (***http://nomachine/test/*** plutôt que ***http://nomachine/test/index.html***). Les noms doivent être séparés par des espaces, et l'ordre dans lequel ils sont saisis conditionne la priorité dans le cas où deux fichiers "par défaut" seraient présents au même endroit.
Pour ne pas retourner systématiquement une erreur 404 signalant une adresse erronée.

**Attention** ! Lorsque aucun nom de fichier n'est fourni, Apache balaye l'ensemble des fichiers du répertoire pour déterminer l'existence ou non de pages "*par défaut*". Donc dans un souci de performance, ne saisissez pas une liste trop longue. Vous pouvez ajouter ***index.php*** devant ***index.html.***

***AccessFileName .htaccess*** ligne 412

Cette directive permet de spécifier le nom d'un fichier qui, s'il est présent dans un répertoire, va permettre de spécifier des règles de sécurité pour l'accès aux autres pages de ce répertoire.

C'est la présence d'un tel fichier qui fait qu'une fenêtre vous demandant votre ***login*** et votre ***mot de passe*** vous est présentée. Un tel fichier est utile, par exemple, pour protéger un répertoire contenant les pages d'administration de votre site.
Le fichier d'accès permet également de "***surcharger***" certaines options définies dans le fichier ***httpd.conf***, c'est-à-dire demander un comportement différent et spécifique pour un répertoire. Cela est intéressant si votre site est hébergé et que vous n'avez pas donc pas la possibilité de jouer sur la configuration de *Apache*.
A noter que ce fichier permet également de spécifier certains options pour ***PHP***. Le choix ou non d'utiliser les fonctionnalités apportées par ce fichier n'est pas évident.
La problématique reste celle du bon équilibre ***« souplesse/performance ».*** Si l'option ***AccessFileName*** existe, chaque fois qu'**Apache** ouvrira un répertoire à la recherche d'un fichier, il cherchera un fichier du nom de ce qui a été spécifié lors de la configuration. Cela ajoute donc une opération et a donc un impact certain sur les performances.

Par contre, on dispose de fonctionnalités appréciables. Au contraire, si cette ligne est commentée (symbole # au début de la ligne), **Apache** n'effectuera pas la recherche d'un fichier. Les performances ne seront pas impactées mais on perd en souplesse.

| ***HostnameLookups Off*** | ligne 490 |
|---|---|

Cette directive conditionne le fait qu'Apache va essayer *(valeur On)* ou non *(valeur Off)* de résoudre ***l'adresse IP*** du client qui se connecte à lui *pour l'écrire dans les journaux d'accès*. Si cette option rend les journaux plus explicites, l'impact sur les performances est généralement trop important. Il vaut donc mieux laisser à *"Off"*

| ***ErrorLog /usr/local/apache/logs/errorJog*** | ligne 499 |
|---|---|
| ***CustomLog /usr/local/apache/logs/accessJog common*** | ligne 525 |

***ErrorLog*** (journal d'erreur) et ***CustomLog*** (journal personnalisé, i.e. journal d'accès) permettent de spécifier le chemin et le nom du fichier contenant respectivement les erreurs d'accès (pages non trouvées, etc.) et les accès réussis. ***CustomLog*** accepte en plus un paramètre qui correspond à un format de journal, c'est-à-dire une plus ou moins grande richesse des informations journalisées. Les formats pour le journal d'accès sont déterminés par la directive ***LogFormat***.

Pour obtenir un maximum d'informations sur les visiteurs, il est bien plus intéressant d'utiliser le mode ***combined*** en lieu et place de ***common***. Mais dans ce cas, le volume des journaux devient vite très importants.

Pour mieux les gérer, il suffit d'avoir recours à un programme permettant "d'éclater" les journaux, le plus facile étant une séparation basée sur la date. L'outil ***cronolog*** est parfaitement indiqué, car il permet de spécifier des noms de fichiers et/ou de répertoires en se basant sur la syntaxe de ***strftime***. Par exemple, ***%Y*** représente l'année courante sur quatre chiffres, ***%m*** le mois courant sur deux chiffres, etc.
Pour connaître l'ensemble des possibilités, faites

```
# man strftime
```

Tout d'abord, commentez la ligne ***CustomLog /usr/local/apache/logs/accessJog common*** (ajout d'un dièse # au début de la ligne) afin de ne plus utiliser le mode standard. Commentez également la ligne contenant ***ErrorLog***.

Ensuite, téléchargez cronolog et décompactez l'archive:

```
# tar xvfz cronolog-1.6.1 .tar.gz
#cd cronolog-1.6.1
```

Préparez la configuration :

```
# ./configure -prefix=/usr/local/cronolog-1.6.1
```

Compilez et installez cronolog :

```
# make install
```

Nous avons choisi de garder les noms de fichier ***accessLog*** et ***errorLog***, mais ceux-ci vont être séparés mensuellement. Un répertoire par année, et un sous -répertoire par mois.

- Pour cela, ajoutez la ligne suivante juste en-dessous de la dernière ligne du type ***CustomLog*** :

```
CustomLog "|/usr/local/cronolog-1.6.1/sbin/cronolog  /usr/local/apache/logs/%Y/%m/accessJog" combined
```

- Ajoutez également une ligne pour le journal des erreurs :

***ErrorLog "|/usr/local/cronolog-1.6.1/sbin/cronolog /usr/local/apache/logs/%Y/%m/error_log[1]".***

**N.B.** : Le caractère avant le chemin vers ***cronolog*** est un pipe ***("|").***

***Alias /icons/ "/usr/local/apache/icons/"*** ligne 585

La directive Alias permet de créer un lien *entre un répertoire dans l'arborescence Web et un répertoire dans l'arborescence du système* de fichiers du serveur.

Ainsi, avec la directive ci-dessus, une requête sur **l'URL *http://nomserveur/icons/test.gif*** retournera le fichier /***usr/local/apache/icons/test/gif***.

**Attention**, si l'alias est défini avec un *slash* à la fin (le symbole "/"), il est nécessaire de taper un slash (ou de spécifier un fichier) dans *l'URL* pour accéder aux données pointées par l'alias.

Par exemple, la directive ***Alias /doc/ "/usr/local/apache/htdocs/manual/"*** permettra d'obtenir le contenu du répertoire manuel si vous saisissez ***http://monserveur/doc/*** ou ***http://monserveur/doc/index.html***

Si vous tapez http://monserveur/doc , **Apache** recherchera un sous-répertoire doc dans l'arborescence standard des documents (voir ***DocumentRoot***). La directive **Alias** apporte une certaine souplesse en évitant de devoir dupliquer certaines données que l'on voudrait rendre accessible.

Cependant, le fait de mettre à disposition des fichiers hors de l'arborescence standard pouvant présenter des risques, il est d'usage d'associer à une directive **Alias** une directive **Directory** afin d'instaurer certaines règles de sécurité.

***AddType application/x-httpd-php3 .php3*** ligne 562-831

- La directive **AddType** permet soit d'ajouter des type ***MIME***, soit d'étendre la liste des extensions de fichiers correspondant à un type ***MIME***.
  L'utilisation la plus courante de cette directive nous concerne directement puisqu'elle permet la prise en compte active par **Apache** des fichiers **PHP**. Nous y reviendrons ultérieurement.

***ErrorDocument 404 /missing.html*** ligne 883

Les directives ***ErrorDocument*** permettent de personnaliser les pages présentées en cas d'erreur. Les paramètres de la directive sont tout d'abord le code ***HTTP*** pour lequel on veut personnaliser, puis soit du texte précédé d'une double quotte (par exemple "Page introuvable), soit une ***URL***. Dans ce dernier cas, l'URL ne doit pas forcément désigner un fichier HTML statique : cela peut être un script Péri (/cgi-bin/erreur404.pl par exemple), une page *PHP*, etc.

**N.B.** : les codes d'erreur les plus courants sont 404 (page introuvable), 500 (erreur interne du serveur) et 403 (accès interdit).

### *f) Virtual Hosts*

Ligne 1017
La rubrique ***Virtual Hosts*** (hôtes virtuels) permet à Apache de gérer plusieurs sites/domaines différents avec le même serveur.

Il existe deux types d'hôtes virtuels : ceux basés sur l'adresse IP et ceux basés sur le nom.

**Attention** toutefois, ces derniers s'appuient sur des spécificités du protocole ***HTTP*** dans ***sa version 1.1***, même si.cela ne devrait plus être un problème à l'heure actuelle car tous les navigateurs récents gèrent ce protocole.
.

Lorsque vous avez terminé les changements dans le fichier de configuration, arrêtez puis redémarrez Apache :

```
# /usr/local/apache/bin/apachectl restart
```

## VI. LANGAGE HTML

- Créer un répertoire au nom de l'utilisateur ***/home/http/toto/index.html***
- Créer la page suivante ***index.html***

```
<html>
        <head>
                <title> Page de toto </title>
        </head>
        <body bgcolor="#E3E3E3" text="#000000">
                <center> <H2> Page de toto ds /home/http/toto </H2></center>
        </body>
</html>
```

Modifier ***httpd.conf***

```
<IfModule mod_userdir.c>
Userdir /home/http
```

Dans votre navigateur lancer : ***127.0.0.1/~toto/index.html***

# PHP

## I. INTRODUCTION

Ce qui allait devenir PHP a été créé en 1995 par le ***Danois Rasmus Lerdorf.*** Son seul et unique but à l'époque était de savoir qui venait lire son ***CV*** sur Internet. Etant à son compte, il envoyait des lettres de motivations à diverses entreprises, et mentionnait l'URL de son site.Il avait donc écrit un script ***CGI*** en langage ***Perl*** qui ajoutait des balises dans le code ***HTML*** qui récupéraient les informations sur les visiteurs.
Pour impressionner ses visiteurs, il choisit de rendre ses statistiques d'accès publiques. Il baptisa cet outil ***PHP-Tools*** (*PHP pour Personal Home Page*) car ce n'était pour lui qu'un outil destiné à sa page personnelle (*home page*). C'est également à ce moment que, suite à plusieurs demandes concernant la disponibilité de son script, il choisit de le diffuser comme ***freeware*** (ou graticiel ;-)...le concept de logiciel libre n'existait pas à l'époque.
Suite à sa recherche d'emploi, *Rasmus* obtint un contrat à l'Université de Toronto où il devait travailler sur un système de connexion à Internet pour les étudiants.
Il devait développer une interface de gestion ***Web*** accédant à la base de données des étudiants hébergée sur un gros système *IBM* et permettant d'autoriser la connexion en fonction du paiement effectué par chaque étudiant.
Cette base devait pouvoir être mise à jour en temps réel. Vu qu'il n'existait alors aucun outil d'interfaçage entre ***HTML*** et base de données, *Rasmus* eut l'idée d'ajouter des balises spécifiques dans les pages ***HTML***, celles-ci étant interprétées par le compilateur ***C***.
Il donna à cet ensemble de balises le nom de ***FI*** pour "***Forms Interpréter***" (interpréteur de formulaire) car elles permettaient de récupérer des informations saisies dans des formulaires puis de les convertir afin de les exporter vers d'autres systèmes. En combinant les fonctionnalités de ***PHP-Tools*** et de ***FI***, *Rasmus* réalisa en 1996 la deuxième version de ***PHP***, ***PHP-FI.***

Bien qu'il lui soit venu l'idée de commercialiser son produit, il s'abstint devant les messages reçus de nombreux programmeurs à travers le monde qui lui envoyait des corrections de bugs et des améliorations. ***PHP*** est dès lors devenu un projet ***Open Source*** et *Rasmus* continue à être l'un des principaux développeurs, même si le moteur d'interprétation de ***PHP*** a été complètement réécrit entre temps.

| | | |
|---|---|---|
| Lieu : | http://www.php.net/downloads.php | **PHP 4.3.1 (tar.gz)** |
| manuel : | http://dev.nexen.net/docs/php/annotee/manuel_tocd.php | |
| manuel français : | http://www.php.net/download-docs.php | |

- Exemple 1-1. Exemple d'introduction

```
<html>
   <head>
      <title>Exemple</title>
   </head>
   <body>

      <?php
      echo "Bonjour, je suis un script PHP!";
      ?>

   </body>
</html>
```

## II. INSTALLATION DE PHP

Pour installer PHP, placez-vous dans le répertoire où vous avez chargé les sources, et décompactez les sources de PHP :

```
tar xvfz php-4.0.6.tar.gz
cd php-4.0.6
```

⇒ **Préparez ensuite la compilation :**

```
./configure \
--with-apxs=/usr/local/apache/bin/apxs\
--enable-inline-optimization \
--enable-debug=no \
--enable-safe-mode \
--enable-calendar \
--enable-ftp \
--enable-sysvsem \
--enable-sysvshm \
--enable-trans-sid \
--with-regex=system \
--disable-static \
--with-regex=system \
--with-mm=/usr/local \
--with-mysql=/usr/local/mysql/  \
--with-freetype-4bit-antialias-hack \
--with-gd-native-tt \
--enable-gd-native-tt \
--enable-freetype-4bit-antialias-hack\
--with-jpeg-dir=/usr \
--with-png-dir=/usr \
--with-tiff-dir=/usr\
--with-zlib-dir=/usr/local \
--with-pdflib=/usr/local/
```

⇒ **Quelques explications sur les options ci-dessus :**

***--with-apxs=/usr/local/apache/bin/apxs ***

indique que PHP va être compilé comme un module dynamique, le paramètre passé correspond au chemin vers le binaire ***apxs***.

***--enable-debug=no***

indique que PHP ne fournira pas d'informations étendues en cas d'erreur. Ce paramètre est à utiliser pour un serveur de production.

***--enable-safe-mode***

permet à PHP de contrôler certains paramètres d'exécution de scripts et assurer une meilleure sécurité.

***--with-regex=system***

indique à PHP d'utiliser la librairie d'évaluation des expressions régulières du système plutôt que celle inclue avec PHP. Validation par le noyau du système.

***--with-mysql=/usr/local/mysql***

spécifie le support de la base de données MySQL.

***--with-gd=/usr/local***

va permettre à PHP de générer des images à la volée en se basant sur la librairie GD compilée précédemment.

***--with-pdflib=/usr/local***

permet à PHP de générer des fichiers PDF à la volée en se basant sur la librairie PDFLib (voir ici pour le détail des fonctions).

Les options ***calendar***, ***ftp***, ***sysvsem*** et ***sysvshm*** ont pour but d'apporter des fonctionnalités supplémentaires à ***PHP***. Elles ne sont en aucun cas obligatoire. Pour plus de détails, vous pouvez consulter la liste des différentes options de compilation, disponible sur le site officiel ***PHP***.

Remarque : Il faut que les package ***flex-5.5.4a-libtiffdevels*** soit installer

Remarque : le paramètre ***--enable-track-vars*** n'est pas spécifié car il est implicite depuis la version 4.0.2 de PHP.

- Lancez la compilation et demandez l'installation de PHP :

```
make
make install
```

## III. REPERTOIRE DE PHP

Le répertoire **/usr/local/apache/conf** contient:

| | |
|---|---|
| ***http.conf*** | le fichier de configuration d'Apache |
| ***mime.types*** | fixe le type de fichier suivant l'extension du dit fichier (.doc=msword,.ps=postscript,...), ça permet au client qui se connecte sur le serveur, de savoir comment interpréter le fichier suivant son extension. |
| ***magic*** | sert pour le module **mod_mime_magic** |
| ***php.ini*** | pour contrôler certains aspects de PHP |

## IV. LE FICHIER DE CONFIGURATION D'APACHE

Le fichier de configuration se trouve dans */usr/local/apache* et se nome ***httpd.***conf

| | |
|---|---|
| ***Servername nom_du_serveur*** | Rajouter ligne 327 |

- Pour activer le support de **PHP** par **Apache**, éditez le fichier ***httpd.conf*** de Apache, recherchez les lignes suivantes et enlever le symbole de commentaire *(#)* au début de chaque ligne :

| | |
|---|---|
| ***AddType application/x-httpd-php .php .php3 .php4 .phtml*** | ligne 818 |
| ***AddType application/x-httpd-php-source –phps*** | |

- Rajouter

| | | |
|---|---|---|
| ***LoadModule php4_module*** | ***libexec/libphp4.so*** | ligne 220 |
| ***AddModule mod_php4.c*** | | ligne 240 |

- Rajouter à la ligne ***DirectoryIndex index.html***

| | |
|---|---|
| ***DirectoryIndex index.html index.htm index.php3 index.php index.php4*** | ligne 409 |

- Si ces lignes ne sont pas décommentées, ***Apache*** va considérer les fichiers portant l'extension ***.php*** comme des fichiers texte simples et va donc afficher le code plutôt que l'interpréter.

- Arrêtez puis redémarrez Apache :

```
/usr/local/apache/bin/apachectl restart
```

Pour apprendre ***PHP*** dirigez-vous vers le site : www.nexen.net
Vous devez ensuite copier le fichier ***php.ini-dist*** dans le répertoire ***/usr/local/lib*** en ***php.ini***.

```
cp /usr/local/src/php-4.0.6/php.ini-dist  /usr/local/lib/php.ini
```

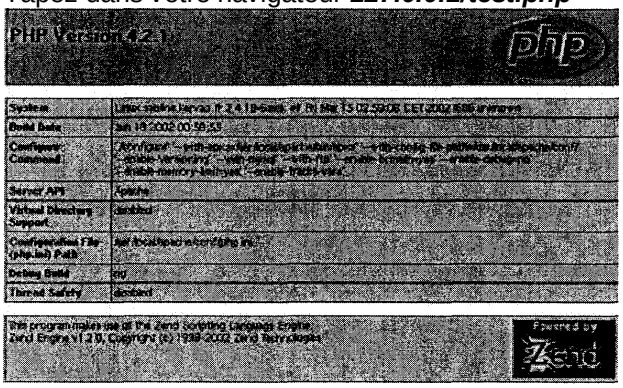
## V. TESTS

### A. Test PHP

Tout d'abord, créez le fichier *test.php* contenant le code suivant dans **/usr/local/apache/htdocs/** :

```
<?
phpinfo ( );
?>
```

Tapez dans votre navigateur **127.0.0.1/test.php**




### B. Test GD

Maintenant, quelques tests des fonctions graphiques fournies par la librairie **GD** (n'oubliez pas de copier le fichier ***arial.ttf*** (la police Arial de Windows) dans le répertoire **/data/fonts**).
Tout d'abord, la création d'une image au format ***PNG***.

Créez un fichier ***imagel.php*** contenant le code suivant :

```
<?php
Header ("Content-type: image/png");
$im = imagecreate (400, 30);
$black = ImageColorAllocate ($im, 0, 0, 0);
$white = ImageColorAllocate ($im, 255, 255, 255);
ImageTTFText ($im, 20, 0, 10, 20, $white, "/data/fonts/arial.ttf", "Testing... Oméga:
&#937;");
ImagePng ($im);
ImageDestroy ($im);
?>
```

Appelez le fichier ***imagel.php*** dans votre navigateur

### C. Test JPEG

Passons maintenant au format ***JPEG***. Créer un fichier ***image2.php*** avec le code suivant :

```
<?php
Header ("Content-type: image/jpeg");
$im = imagecreate (400, 30);
$black = ImageColorAllocate ($im, 0, 0, 0);
$white = ImageColorAllocate ($im, 255, 255, 255);
ImageTTFText ($im, 20, 0,10, 20, $white, "/data/fonts/arial.ttf", "Testing... Oméga:
&#937;");
ImageJpeg ($im);
ImageDestroy ($im);
?>
```

Appelez le fichier ***image2.php*** dans votre navigateur

## VI. DETAIL DU FICHIER DE CONFIGURATION D'APACHE / PHP

Le fichier de conf **d'Apache** se trouve sous **/usr/local/apache** et se nomme ***httpd.conf*****,** voici les points que je juge important dans le fichier:
(...)

**# —-.,.——.....— SERVER CONFIGURATION ————————————**

*# **ServerType** is either **inetd**, or **standalone**.*
*# vous pouvez soit lancer Apache par le super daemon **inetd** (config dans **/etc/inetd.conf**)* soit tout seul avec un script dans ***/etc/rc.d/init.d***, à noter *qu'avec la première méthode vous pouvez contrôler l'accès avec les **TCP_Wrappers** (hosts.deny, hosts.allow)*
*# voir la note en bas du paragraphe pour les avantages de l'un ou de l'autre*
*# moi j'ai choisi lancement en **standalone** (voir paragraphe suivant)*

```
ServerType standalone
(...)
```

*# Do NOT add a slash at thé end of thé directory path. Répertoire racine d'Apache*

```
ServerRoot "/usr/local/apache"                                    534
(...)
```

*# **PidFile**: Thé file in which thé server should record its process identification number when it starts.*
*# C'est dans ce fichier qu'on trouvera l'identification du process d'Apache c'est utile pour ne pas*
*# lancer deux fois Apache par exemple **PidFile/usr/local/apache/logs/httpd.pid***
*(...)*

*# Limit on total number of servers running, Le., limit on thé number of clients who can*
*# simultaneousiy connect — if this limit is ever reached, clients will be LOCKED OUT, so it should*
*# NOT BE SET TOO LOW. # It is intended mainly as a brake to keep a runaway server from taking*
*# Unix with it as it spirals down...*
*# ici on fixe le nombre de clients max qui peuvent se connecter à un moment* donné

```
MaxClients150
(...)
```

*# Suit ensuite une liste de **modules**, ceux avec un # devant ne seront pas inclus*

```
LoadModule vhost_alias_module libexec/mod_vhost_alias.so
LoadModule env_module    libexec/mod_env.so
LoadModule config_log_module libexec/modJog_config.so
LoadModule mime_magic_module libexec/mod_mime_magic.so
LoadModule mime_module   libexec/mod_mime.so
LoadModule negotiation_module libexec/mod_negotiation.so
LoadModule status_module   libexec/mod_status.so
LoadModule info_module    libexec/mod_info.so
(...)
```

Pour chaque module on doit maintenant la routine **AddModule**

```
ClearModuleList
AddModule mod_vhost_alias.c
AddModule mod_env.c
AddModule mod_log_config.c
AddModule mod_mime_magic.c
AddModule mod_mime.c
AddModule mod_negotiation.c
AddModule mod_status.c
AddModule mod_info.c
AddModule mod_include.c
AddModule mod_autoindex.c
AddModule mod_dir.c
AddModule mod_cgi.c
(...)
```

# Port: Thé port thé ***standalone*** listens to.
# For ports < 1023, you will need ***httpd*** to be run as root initially.

# Numéro de port pour le serveur, traditionnellement à 80

```
Port 80
(-)
```

# On lance initialement **httpd** en tant que **root**, puis immédiatement
# c'est l'utilisateur **nobody** (groupe nobody) qui en devient le propriétaire
# ainsi s'il y a une faille dans Apache, le hacker au lieu de devenir **root** devient **nobody** avec les
# droits qui vont avec
# pour vérifier que **nobody** est bien le proprio : ***ps aux | grep httpd***

```
User nobody
Group nobody
(...)
```

# ***ServerAdmin***: Your address, where problems with the server should be
# e-mailed. This address appears on some server-generated pages, such as error documents.
# En cas de problème un émail sera envoyé au ***webmaster***, mettez donc ici l'adresse du webmaster

```
ServerAdmin david@cybergate-paradize.fr
```

# ***DocumentRoot***: The directory out ofwhich you will serve your
# documents. By default, ail requests are taken from this directory, butsymbolic links and aliases
# may be used to point to other locations.
# C'est dans ce répertoire qu'on va trouver la page d'accueil ***d'Apache***

```
DocumentRoot "/usr/local/apache/htdocs"
(...)
```

# ***UserDir***: The name ofthe directory which is appended onto a user's homedirectory if a –user
# request is received.
# Chaque utilisateur pourra mettre ses pages dans sa ***homedirectory*** dans un répertoire
# ***public_html***, les pages seront accessibles à l'URL ***http://localhost/~user***

```
<IfModule mod_userdir.c>
UserDir public_html
</IfModule>
(...)
```

# ***ErrorLog***: The location ofthe error log file. If you do not specify an ***ErrorLog*** directive within a
# <VirtualHost> container, error messages relating to that virtual host will be logged hère. If you *do*
# define an error logfile for a <VirtualHost> container, that host's errors will be logged there and not
# here.
# L'emplacement du fichier de ***log*** peut ne pas vous convenir, si vous voulez le placer dans
# ***/var/log/httpd*** il faut modifier ici

```
ErrorLog/usr/local/apache/logs/error_log
```

#The location and format ofthe access ***logfile*** (Common Logfile Format). If you do not define any
# access logfiles within a <VirtualHost> container, they will be logged here. Contrariwise, if you *do*
# define per-<VirtualHost> access logfiles, transactions will be logged therein and *not* in this file.
# *De même pour l'emplacement du fichier de log des accès à apache c'est cette variable qu'il faut*
# *modifier*

```
CustomLog /usr/local/apache/logs/access_log common
```

# ***DirectoryIndex***: Name of the file or files to use as a pre-written HTML directory index. Separate
multiple entries with spaces.
# liste des fichiers d'entrée des pages

```
<IfModule mod_dir.c>
DirectoryIndex index.html index.htm index.php3 index.php index.php4
</IfModule>
```

# priorité dans les langages à utiliser, mettez ***fr*** en premier

```
IfModule mod_negotiation.c>
LanguagePriority fr en da ni et de el itja pi pt pt-br ltz ça es sv
</IfModule>
```

# Apache peut se comporter comme un ***proxy*** comme ***squid*** par défaut cette fonction n'est pas activée
# Proxy Server directives. Uncomment thé following line to enable thé proxy server:
# ProxyRequests On
# To enable the cache as well, edit and uncomment the following lines:

```
# CacheRoot /var/cache/httpd
# CacheSize 5
# CacheGcInterval 4
# CacheMaxExpire 24
# CacheLastModifiedFactor 0.1
# CacheDefaultExpire 1
# NoCache a_domain.com another_domain.edujoes.garage_sale.com
```

**NOTE** "Anciennement" on trouvait un fichier **srm.conf** et **access.conf,** ils peuvent maintenant complètement être intégrés dans **httpd.conf**

⇒ **Lancement automatique de l'application**

On prendra le fichier ***apachectl*** se trouvant sous **/usr/local/apache/bin** et on le placera sous **/etc/rc.d/init.d**, et on le renommera **httpd**

```
cp /usr/local/apache/bin/apachectl /etc/rc.d/initd/httpd
```

Puis on éditera le fichier **/etc/rc.d/rc. local** et on ajoutera en fin de fichier la ligne suivante :

```
./etc/rc.d/initd/httpd start
```

## VII. LES PAGES WEB UTILISATEURS

Le problème avec le répertoire ***public_html*** des utilisateurs et qu'il faut mettre ***755*** au niveau de la home directory, ce qui est particulièrement gênant au niveau sécurité. Vous pouvez spécifier que chaque utilisateur doit créer ses pages sous **/home/http/login-utilisateur** en écrivant pour la variable ***UserDir***

```
UserDir    /home/httpd
```

Ainsi pour l'utilisateur **toto** quand vous taperez comme **URL >** ***http://serveur-apache/~toto*** , ***apache*** ira chercher le fichier **index.htm** sous **/home/httpd/toto .**
On peut aller plus loin en spécifiant un répertoire particulier, ***/home/httpd/toto/html*** par exemple, en écrivant:

```
UserDir     /home/httpd/*/html
```

## VIII. LES ALIAS

Si vous ne voulez pas mettre en place un serveur ***DNS***, vous avez un moyen plus simple, ***les alias***. Concrètement, votre serveur s'appelle **obelix** , vous voulez rendre accessible les fichiers ***html*** se trouvant sous **/usr/doc/html** , les utilisateurs devront taper dans leur navigateur préféré: http://obelix/doc.

- Pour cela dans votre fichier **/etc/httpd/conf/*httpd.conf***, vous allez rajouter:

```
Alias /icons/ "/usr/local/apache/icons/"
Alias /doc "/usr/doc/html/"
```

**NOTE** Si vous mettre***/doc/***à la place de ***/doc*** dans l'URL il faudra taper **http://obelix/doc/,** si vous omettez le dernier /, vous aurez une erreur

## IX. CONFIGURATION AVANCEE / PROTECTION D'UNE PAGE

### *a) Principe de l'authentification*

- Le système d'authentification est déclenché lorsque le serveur détecte dans le répertoire contenant le document à transmettre la présence du fichier ***.htaccess*** dont la lecture lui indiquera quel est le type de protection en service dans ce répertoire.
- Ce système d'authentification est très souple puisqu'il va permettre de gérer différents répertoires selon des critères d'accès différents.

Il faut décommenter maintenant les lignes suivantes dans le fichier ***httpd.conf*****:**

| | |
|---|---|
| ***AccessFileName .htaccess*** | ligne 412 |

### *b) Authentification par mot de passe*

Pour ce faire, la première étape va constituer en la création de la base de données qui est un fichier de nom ***.htpasswd*** et qui va contenir les caractéristiques des personnes autorisées à accéder aux documents.
Dans un serveur de type NCSA un utilitaire est fourni pour créer et gérer cette base de données. Il s'agit du programme ***htpasswd*** accessible directement sous Unix.

- On peut créer d'abord un répertoire **(basedenoms** par exemple) qui va contenir la base de données et on lui met des droits de lecture pour tout le monde. Notez bien que ce répertoire peut se situer n'importe où sur le site.

```
mkdir basedenoms
chmod 755 basedenoms
```

- Après s'y être déplacé (***cd basedenoms***), on crée ensuite simultanément (option -c) la base de données ***.htpasswd*** et le premier utilisateur avec son mot de passé le programme se situe dans ***/usr/local/apache/bin***

```
htpasswd -c .htpasswd david
Adding password for david.
New password:
Ré-type new password:
```

Remarquez que dans cet exemple le fichier **.htpasswd** se trouvera donc précisément dans le répertoire **basedenoms.**
**Remarque :** sur pc, la commande ***htpasswd*** n'est pas livrée en standard avec le système.

- On peut ensuite ajouter d'autres utilisateurs (cette fois-ci sans l'option de création -c)

```
./htpasswd .htpasswd stef
Adding user stef
New password:
Ré-type new password:
```

- Pour contrôle, vous pouvez visualiser le contenu de la base :

```
% cat .htpasswd
david:bp3rCaQn8clSw
stef:D1.766H0012hA
```

Vous pouvez constater que ce fichier contient simplement les noms des utilisateurs suivis de leur mot de passe crypté.

- Il est à remarquer qu'afin de "diminuer" leur lisibilité au moment du passage sur le réseau, les mots de passe sont ***encodés***. Mais il ne s'agit en aucun cas d'un encryptage sécurisé.
- Par ailleurs, exécuter une seconde fois cette commande pour un même utilisateur ne fera que changer son mot de passe.
- Si vous désirez retirer un utilisateur, il suffit d'éditer le fichier **.htpasswd** et de supprimer la ligne correspondant à cet utilisateur.

**Remarque** : La création et la mise à jour du fichier ***.htpasswd*** peuvent également être effectués par des scripts ou des programmes ***CGI***, la seule difficulté résidant dans le codage du mot de passe pour qu'il puisse être reconnu par le serveur.

Pour cela, les langages de script ou d'écriture de programmes CGI offrent en général une fonction qui permettra de le faire.
Par exemple, dans un script ***php***, vous pouvez utiliser la fonction ***crypt(chaine).***
Une fonction analogue existe également en C.

- On se positionne ensuite dans le répertoire contenant les fichiers ***HTML*** à protéger.
- (dans notre exemple ces fichiers ne seront pour le moment éventuellement accessibles qu'aux utilisateurs david et stef).

- On crée dans ce répertoire le fichier ***.htaccess*** qui devra contenir les lignes suivantes :

```
AuthName message
AuthUserFile /usr/local/bin/www/httpd_1 .S/basedenoms/.htpasswd
AuthGroupFile /dev/null
 AuthType Basic
<limitGET>
require valid-user
</Limit>
```

  - *Le texte situé derrière le nom* **AuthName** *correspond au message que vous désirez voir afficher dans la boîte de dialogue lors de la demande d'authentification.*
  - *Pour la ligne* **AuthUserFile** *vous devez donner le chemin complet du fichier, incluant le chemin jusqu'à votre répertoire racine (obtenu avec la commande pwd).*
  - *La ligne suivante* ***(require)*** *signifie qu'aucune distinction n'est faite parmi les membres de la liste.*
- Si vous vouliez choisir, parmi les membres de la liste figurant dans ***.htpasswd***, ceux auxquels vous voulez donner les droits d'accès, il faudrait les citer nommément :

```
AuthName message
AuthUserFile /usr/local/bin/www/httpd_1 .S/basedenoms/.htpasswd
AuthGroupFile /dev/null

<limit GET>
require user durand dupont dupond
</Limit>
```

Dans cet exemple, seuls **durand**, **dupont** et **dupond** pourront passer la barrière !

- C'est également grâce à la directive ***require*** qu'il est aussi possible de gérer des groupes d'utilisateurs. Pour ce faire il suffit par ailleurs de créer un fichier **.htgroup** (toujours dans un répertoire de son choix). Ce fichier sera structuré comme dans l'exemple suivant qui défini trois groupes (non forcément disjoints) :

```
admin-grp: dubois etienne gaillard
systm-grp: perrot dumas
visit-grp: dufour
```

- Chaque répertoire pourra être ensuite ouvert au(x) groupe(s) concerné(s) comme l'indique par exemple le fichier ***.htaccess*** de l'exemple suivant autorisant les groupes **admi-grp** et **systm-grp** à accéder aux fichier HTML contenus dans le même répertoire

```
AuthName message
AuthUserFile /usr/local/bin/www/httpd_1 .S/basedenoms/.htpasswd
AuthGroupFile /usr/local/bin/www/httpd_1.3/basedenoms/.htgroup
AuthType Basic
<limit GET>
require group admin-grp
require group systm.grp
</Limit>
```

- ***<limit>*** est un bloc contenant des sous-directives permettant de définir les droits d'accès associés à une ou plusieurs méthodes d'accès (GET, PUT).

- La directive **require** est utilisée dans le cas d'accès par utilisateur et mot de passe et spécifie quels sont les utilisateurs ou groupes ayant accès aux pages du répertoire où se trouve le fichier ***.htaccess*.**

Elle peut avoir les formes suivantes :
***--- require valid-user :*** tous les membres de la liste ***.htpasswd*** sont autorisés à lire les fichiers du répertoire
***-- require group groupel groupe2 :*** seuls les membres de la liste générale qui font aussi partie des groupes groupel et groupe2 sont autorisés.
***--require user usagerl usager2 usagers :*** seuls usagerl, usager2 et usagers auront les droits, à condition qu'ils soient aussi dans ***.htpasswd***

**Remarque :** Si vous souhaitez établir une hiérarchie de répertoires pour lesquels les droits se restreignent au fur et à mesure que l'on descend dans la hiérarchie, il suffit que les répertoires attenant à chacun des répertoires concernés fasse référence au même fichier des mots de passe (***.htpaswd).***
Dans ce cas, pourvu qu'un utiisateur n'aille que dans les lieux qui lui sont permis, son mot de passe ne lui sera demandé qu'une unique fois !

### *c) Authentification par le réseau*

- Dans cette méthode on va autoriser ou interdire l'accès à un groupe d'utilisateurs appartenant à un même domaine.
- Dans les exemples suivants on offrira d'abord l'accès aux pages ***HTML*** exclusivement aux utilisateurs venant du domaine ***.ch*** et du domaine ***.enst.fr*** puis on interdira l'accès uniquement aux utilisateurs de la machine ***ulysse.enst.fr.***

Ici, seule la présence du fichier ***.htaccess*** est nécessaire. Son contenu est bien sûr différent du cas de l'authentification par nom et mot de passe.

Deux possibilités sont donc offertes :

- On refuse tous les accès sauf ceux qui sont précisés

```
AuthUserFile /dev/null
AuthGroupFile /dev/null
AuthName AccesRestreint
AuthType Basic
<limit GET>
order deny,allow
deny from all
allow from .ch
allow from .enst.fr
</Limit>
```

- On accepte tous les accès sauf ceux qui sont précisés

```
AuthUserFile /dev/null
AuthGroupFile /dev/null
AuthName InterditPartiel
AuthType Basic
<limit GET>
order allow,deny
allow from all
deny from ulysse.enst.fr
</Limit>
```

### *d) Combinaison des deux méthodes*

- Il est possible de combiner accès par mot de passe et accès par sous-domaine. Dans ce cas, la présence des deux fichiers **.htpasswd** et **.htaccess** est nécessaire. Le contenu de ce dernier est bien sûr différent de celui des cas précédents.

Deux possibilités sont offertes :

- On autorise les accès depuis un certain sous-domaine plus des mots de passe ailleurs

```
AuthUserFile /usr/local/bin/www/httpd_1 -S/basedenoms/.htpasswd
AuthGroupFile /dev/null
AuthName AccesRestreint
AuthType Basic
<limit GET>
order deny,allow
allow from .enst.fr
require user martin
satisfy any
</Limit>
```

- Dans cet exemple, sont autorisées les personnes connectées depuis le sous-domaine ***.enst.fr*** ainsi que la personne de nom **martin** (citée dans le fichier ***.htpasswd***) qui peut se connecter d'un tout autre endroit.
- C'est la clause ***satisfy any*** qui permet l'une ou l'autre des deux autorisations.
- On autorise les accès depuis un certain sous-domaine en faisant de plus intervenir des mots de passe

```
AuthUserFile /usr/local/bin/www/httpd_1 .S/basedenoms/.htpasswd
AuthGroupFite /usr/local/bin/www/httpd_1 .S/basedenoms/.htgroup
AuthName AccesRestreint
AuthType Basic
<limitGET>
order deny.allow
deny from all
allow from .enst.fr
require groupintranet
satisfy all
</Limit>
```

- Dans cet exemple, sont autorisés les personnes connectées depuis le sous-domaine ***.enst.fr*** à condition de faire partie du groupe ***intranet*** (défini dans ***.htgroup***). C'est la clause ***satisfy all*** qui permet la combinaison des deux conditions.

Cette possibilité est particulièrement utile dans le cas où on aurait défini deux groupes non forcément disjoints et qu'on souhaite ne donner l'accès qu'à l'intersection des deux groupes, sans remettre en cause la constitution propre de chaque groupe.

### *e) Propagation de la protection*

Lorsque l'on protège un répertoire tous les sous-répertoires sont automatiquement protégés.
Exemple d'utilisation sur Emma
Création de la directory contenant les mots de passe :

```
mkdir/infres/emma/infgl/danzart/basedenoms
```

Création de la base et du premier utilisateur :

```
htpasswd /infres/emma/infgl/danzart/basedenoms/.htpasswd -c utilisateuri
```

Création ou modification d'un utilisateur :

```
htpasswd /infres/emma/infgl/danzart/basedenoms/.htpasswd utilisateur2
```

### *f) Protéger ensuite un répertoire :*

- Placer le fichier **.htaccess** dans le répertoire devant être protégé. L'éditer en mettant les directives de restriction souhaitées.

*Cas d'erreur fréquent !*
Si les fichiers de .***htpasswrd*** ou ***.htaccess*** ne sont pas autorisés en lecture pour tout le monde ***(o+r***), le serveur n'y aura pas accès, et votre protection sera inefficace.

```
<Directory /home/*/public_html>
AllowOverride Fileinfo AuthConfig Limit
Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
<Limit GET POST OPTIONS PROPFIND>
Orderallow,deny
Allow from all
</Limit> .
<LimitExcept GET POST OPTIONS PROPFIND>
Orderdeny.allow
Deny from all
</LimitExcept>
</Directory>
```

- Ces lignes autorisent l'emploi du fichier ***.htaccess*** dans les pages de vos utilisateurs *(répertoire /home/*/public_html)*

Maintenant de votre navigateur préféré quand vous allez rentrer comme URL http://obelix/~david/reserve , vous aurez une fenêtre popup qui vas'ouvrir vous demandant de rentrer votre nom d'utilisateur et le mot de passe préalablement rentré.

Notez que pour que quelqu'un ne puisse jeter un coup d'œil dans les fichiers **.htaccess** de vos utilisateurs, le fichier **httpd.conf,** contient la directive suivante:

```
<Files ~ "^\.hf">
Order allow,deny
Deny from all
Satisfy Ail
</Files>
```

## X. LES HOTES VIRTUELS

On peut mettre en place des hôtes virtuels, en d'autres termes un utilisateur pour un même serveur Apache croira en voir plusieurs. Exemple, soit votre serveur Apache **obelix** (adresse IP **192.168.24.56),** votre domaine **breiziand.bz,** on va créer les hôtes virtuels ***www.asterix.cybergate-paradize.fr*** et ***www.idefix.cybergate-paradize.fr*** qui vont pointer chacun vers un endroit différent du disque (respectivement ***/usr/local/asterix*** et***/usr/local/idefix*** chacun contenant des pages html).

Dans le fichier ***httpd.conf*** le module est déjà chargé :

```
LoadModule vhost_alias_module libexec/mod_vhost_alias.so
AddModule mod_vhost_alias.c
```

On va rajouter tout à la fin du fichier:

```
NameVirtualHost 192.168.24.56
<VirtualHost 192.168.24.56>
ServerName obelix.cybergate-paradize.fr
DocumentRoot/usr/local/apache/htdocs
ErrorLog logs/obelix-error_log
TransferLog logs/obelix-access_log
</VirtualHost>

<VirtualHost 192.168.24.56>
ServerName www.asterix.cybergate-paradize.fr
DocumentRoot /usr/local/asterix
ErrorLog logs/asterix-errorJog
TransferLog logs/asterix-access_log
</VirtualHost>

<VirtualHost 192.168.24.56>
ServerName www.idefix.cybergate-paradize.fr
DocumentRoot /usr/local/idefix
ErrorLog logs/idefix-error_log
TransferLog logs/idefix-access_log
</VirtualHost>
```

Relancez Apache en tapant:

```
/etc/rc.d/init.d/httpd restart
```

Maintenant nous allons créer nos hôtes **asterix** et **idefix,** pour cela vous avez deux méthodes:

- rajouter ***www.asterix.cybergate-paradize.fr*** et ***www.idefix.cybergate-paradize.fr*** dans ***/etc/hosts*** sur la même ligne que votre serveur Apache ( **obelix** dans notre exemple).

```
192.168.24.56 obelix obelix.breiziand.bz www.asterix.cybergate-paradize.fr
www.idefix.cybergate-paradize.fr
```

Normalement si vous faites un ping sur **www.idefix.cybergate-paradize.fr** ça devrait marcher, pour les postes clients il faudra rajouter la même ligne dans le fichier **hosts** (non nécessaire).

- si vous disposez d'un serveur DNS sur votre machine, au niveau de votre config DNS dans votre fichier **breiziand.bz** qui se trouve sous ***/var/named*** vous devez rajouter tout à la fin:

```
www.asterix A 192.168.24.56
www.idefix A 192.168.24.56
```

Relancez le DNS en tapant:

```
/etc/rc.d/init.d/named restart
```

Pour tester tapez dans un shell:

```
ping www.asterix.cybergate-paradize.fr
```

Maintenant dans le champ URL de votre navigateur préféré:

```
http://www.asterix.cybergate-paradize.fr
```

Et là, normalement vous devriez voir s'afficher la page que vous avez placé sous ***/usr/local/asterix***

# ☯ GESTION DE BASES DE DONNEES AVEC MYSQL

⇒ **Tests de fonctionnement avec MySQL**

On suppose que vous avez installé, configuré **MySQL** et créé les deux exemples de la page **MySQL.** On suppose aussi que le serveur s'appelle obelix et l'utilisateur **david.**
Voici une page écrite en **PHP** qui va accéder à la base de donnée essai et à sa table **coord.**

```
<?
$serveur="localhost";
$login="david";
$pass="mot-de-passe";
$base="essai";
$table="coord";
$id=MYSQL_CONNECT($serveur,$login,$pass);
mysql_select_db($base);
$nom="parize";
$prenom="david";
$email="david.parize@yahoo.fr";
$query="INSERT INTO Stable VALUES('$nom','$prenom','$email')";
$result=mysql_query($query,$id);
écho "Saisie terminée";
?>
```

Placer ce script dans ***~/public_html*** et appeler le ***bd1.php***

Dans votre navigateur préféré, dans le champ URL saisissez :

```
http://obelix/~david/bd1 .php
```

A priori y a pas grand chose qui s'est passé, maintenant connecter vous à votre base **essai** dans un shell

```
[david@obelix david]$ mysql -u david-p essai

Enter password:
Welcome to thé MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 10to server version: 3.22.32
Type 'help' for help.

mysql> SELECT * FROM coord;
+-..-+—.—.+—..—.—.——.—+
| nom  | prénom | émail              |
| parize| david | david.parize@yahoo.fr |
+-.—+—.—.+..—.———..-.—+
1 row in set (0.00 sec)
```

- C'est bon ça fonctionne. Passons à un exemple plus pointu, on va entrer les informations concernant vos visiteurs dans une base **MySQL** ,

créer la table telle que décrite dans l'exemple 2 de la page **MySQL,** créer maintenant le script **PHP.**

```
<?
$page=getenv("HTTP_REFERER");
$ip=getenv( "REMOTE_ADDR");
$host=gethostbyaddr($ip);
$d = date("d/m/Y H:i:s");
$expl=getenv("HTTP_USER_AGENT");
$serveur="localhost";
$login="david";
$pass="mot»de-passe";
$base="essai";
$table="ref;
$id=MYSQL_CONNECT($serveur,$login,$pass);
mysql_select_db($base);
$query="INSERT INTO $table VALUES('$d','$host','$ip','$expl','$page')";
$result=mysql_query($query,$id);
écho "$d $host($ip) $expl $page";
?>
```

- Nommez ce script **bd2.php** et placez le dans **~/public_html** .
- Dans votre navigateur préféré tapez dans le champ URL

```
http://obelix/~david/bd2.php
```

Vous devriez voir la date, le nom de votre machine avec son adresse IP et des infos sur votre OS et votre navigateur. A présent connectons nous à la base:

```
[david@obelix david]$ mysql -u david-p essai

Enter password:
Welcome to thé MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 10 to server version: 3.22.32
Type 'help' for help.

mysql> SELECT * FROM ref;

+——————— +•——_—_——+->——— +—_———— +—— +
| date          | host          | ip        | os              |
page |
| 24/04/2000 08:34:05 | asterix.armoric.bz 1192.168.24.56 | Mozilla/4.61 [en] (X
"I"                  +                  —+            J-                    •L      J.
1 row in set (0.00 sec)
```

C'est bon le visiteur a bien été pris en compte.
Maintenant que vous savez comment Apache fonctionne avec **MySQL** et **PHP,** laissez libre cours à votre imagination.

# ☯ ADMINISTRATION DES BASES MYSQL AVEC PHPMYADMIN

**phpMyAdmin** est un ensemble de scripts PHP qui permet d'administrer des bases **MySQL** à partir d'un navigateur. Vous pouvez **le** récupérer à l'URL
www.phpwizard.net/phpMvAdmin

. En détail phpMyAdmin permet de:

- créer et supprimer des bases de données,
- éditer, ajouter ou supprimer des champs,
- taper des commandes SQL,
- gérer les clés de champs,

…/…

## I. INTRODUCTION

Logiciel de création de base de données sur ***MySql*** avec Interface Graphique.

Lieu : http://www.phpmyadmin.net/ ***phpMyAdmin-2.4.0-php.tar.gz***
Manuel : http://www.phpmyadmin.net/documentation/

L'archive se présente sous la forme d'un tarball **phpMyAdmin-2.4.0-php.tar.gz** ,pour décompresser:

## II. INSTALLATION

```
tar xvzf phpMyAdmin-2.4.0-php.tar.gz /usr/local/apache/htdocs
```

- Cela va créer dans le répertoire de travail un répertoire **phpMyAdmin-2.4.0.**
- Dans ce répertoire on va éditer le fichier ***config.inc.php3*.**
- On doit d'abord indiquer l'URL pour atteindre ***phpMyAdmin*.**

## III. CONFIGURATION

- éditer le fichier ***config.inc.php***
- ajouter

```
$cfg['Pmabsolute URI']='127.0.0.1/phpmyadmin{          ligne 26
```

- Lancer le navigateur

```
127.0.0.1/phpmyadmin/index.php
```

```
$cfgPmaAbsoluteUrl = 'URL pour atteindre phpMyAdmin (voir plus bas)';
```

On définit l'utilisateur pour accéder à la base **MySQL**

```
$cfgServers[$i]['user']     = 'david';   // MySQL user
$cfgServers[$i]['password']   = 'mot-de-passe-en-clair';
```

Et maintenant pour avoir la version française, dans le même fichier au lieu de:

```
$cfgDefaultLang = 'en';
```

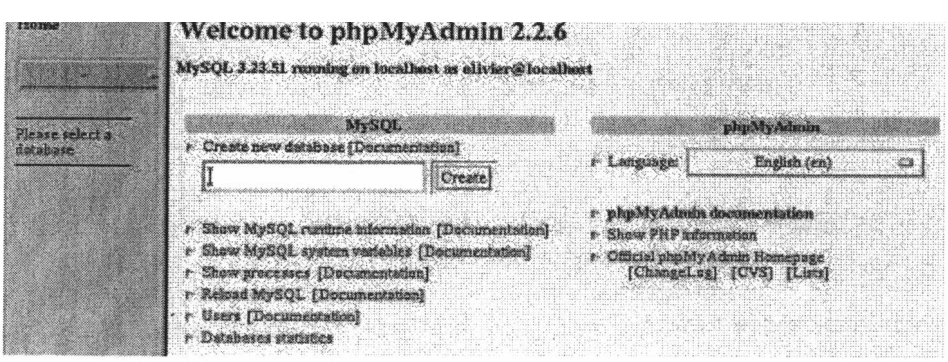
On va mettre

```
$cfgDefaultLang = 'fr';
```

Maintenant on doit rendre accessible le répertoire **phpMyAdmin** d'une page web, pour cela deux solutions:

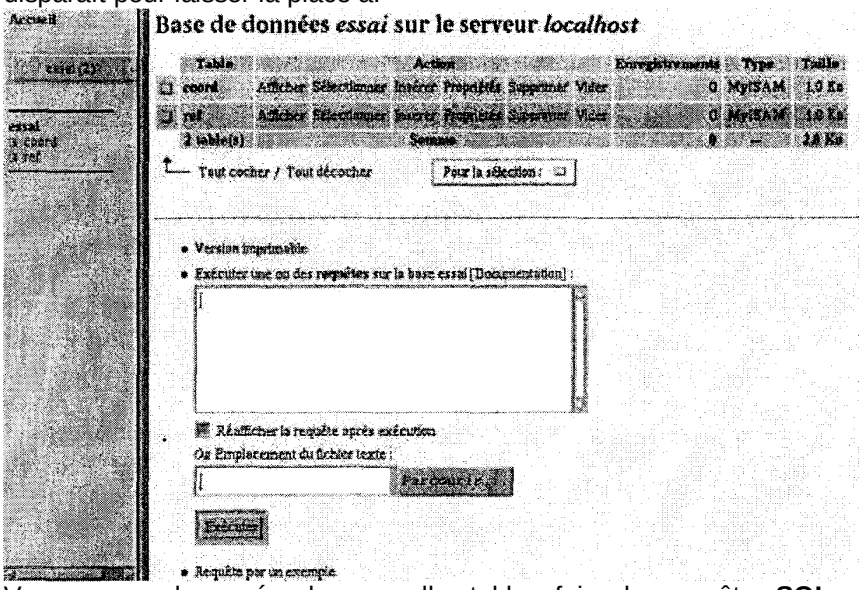
- Solution simple :
  placer ***phpMyAdmin*** dans ***/usr/local/apache/htdocs*** et au niveau de la page d'accueil d'apache faire un lien vers ***/usr/local/apache/htdocs/phpMyAdmin-2.4.0/index.php3***
- Solution préconisée :
  créer un hôte virtuel pointant vers ***./phpMyAdmin-2.4.0*** qu'on appelera ***www.sql.cybergate-paradize.fr*** .
- Modifiez maintenant la variable ***$cfgPmaAbsoluteUri***

**NOTE** Si ça vous gène que n'importe qui d'un navigateur puisse aller dans lerépertore **phpMyAdmin,** mettez y des restrictions d'accès avec un fichier **.htaccess.**
Avec la solution hôte virtuel, à partir d'un navigateur quand on sélectionne **www.sql.cybergate-paradize.fr** on tombe sur une fenêtre avec frame avec à gauche la liste des bases de données disponibles et à droite, le menu suivant:




Pour travailler sur une base de données particulières il suffit de la sélectionner dans le choix déroulant à gauche, on retrouve d'ailleurs notre base **essai,** pour en créer une autre il suffit de choisir **Create new database.**

Si on sélectionne **essai** par exemple et plus particulièrement la table *coord*, le frame de droite disparaît pour laisser la place à:




Vous pouvez donc créer des nouvelles tables, faire des requêtes **SQL,** etc.

# ☯ SCRIPT CGI

Pour activer les scripts CGI, le fichier ***httpd.conf*** est déjà configuré pour, on y trouve notamment la ligne qui indique où trouver les scripts:

```
ScriptAlias /cgi-bin/ /usr/local/apache/cgi-bin/
```

## I. PERL

### A. Installation de mod_perl

- Placez vous dans le répertoire des sources de mod_perl.
  Entrez:

```
perl Makefile.PL USE_APXS=1 \
WITH_APXS=/usr/local/apache/bin/apxs EVERYTHING=1
```

- où /***usr/local/apache/bin/apxs*** est le chemin vers le fichier ***apxs*** de votre installation d'apache
  Par les options ***USE_APXS=1 et WITH_APXS=/usr/local/apache/bin/apxs***, la compilation se fait correctement vis à vis de mod_ssl.
- Lancez ensuite:

```
make
make install
```

### B. Script

Le but de l'exercice est de créer un ***script perl*** CGI qui va traiter un formulaire quelconque d'une page HTML. Vous allez créer votre script pérl sous ***/usr/local/apache/cgi-bin*****,** et le nommer *form.pl*, voici son contenu:

```
#!/usr/bin/perl
use CGI;
$html=new CGI;
print $html->header;
print"<HTML>\n";
print "<HEAD>\n";
print "<TITLE>Premier script CGI perl</TITLE>\n";
print "</HEAD>\n";
print "<BODY>\n";
print "<H1>Traitement du formulaire</H1>\n";
print "Nom :";
print $html->param('nom');
print "<p>\n";
print "Email :";
print $html->param('email');
print "<p>\n";
print "Commentaire:";
print $html->param('comment');
print "</BODY>\n";
print "</HTML>\n";
```

Donner les droits qui vont bien avec ce fichier:

```
chmod 755 form.pl
```

En tant qu'utilisateur standard **(david** dans notre exmple), créer maintenant le fichier HTML suivant que vous appelerez ***formulaire.htm***

```
<html>
<body>
<h2>Formulaire</h2>
<form action="http://obelix/cgi-bin/form.pl" METHOD=GET>
Nom: <input type="text" name=nom size=20><br>
Email: <input type="text" name=email size=30><br>
Commentaire: <input type="text" name=commentsize=100><br>
<input type=submit value="Envoyer"> <input type=reset value="remettre à
zéro">
</form>
</body>
</html>
```

Voilà maintenant quand vous allez accéder à ***http://obelix/~david/formulaire.htm*** **,** vous allez avoir une page du style:
Haut du formulaire

**Nom:**
**Email:**
**Commentaire:**


***Bas du formulaire***

En appuyant sur **Envoyer** ça va déclencher l'exécution du script **CGI** péri, qui va provoquer l'affichage des valeurs précédemment saisies.

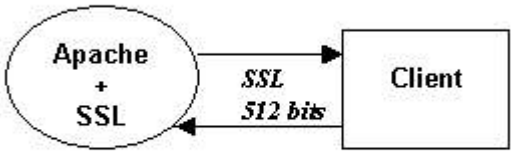
## II. SCRIPT JAVA

# ☯ OPENSSL

## I. INTRODUCTION

***Openssl*** est un serveur ***SSL*** permettant de crypter les informations entre le client et le serveur.

Lieu : http://www.openssl.org/source/ openssl-0.9.7.tar.gz openssl-0.9.6e.tar.gz
Manuel : http://www.openssl.org/docs/apps/openssl.html




## II. INSTALLATION

- L'installation est très simple

```
tar xvzf openssl-0.9.7..tar.gz
cd openssl-0.9.7
./config
make
make test
make install
```

L'installation est terminée

## III. CREER UN CERTIFICAT DE SECURITE

On commence par installer une clé privée. J'ai choisi de ne pas la protéger par mot de passe pour éviter la complication de la lecture par apache d'un mot de passe à chaque démarrage ou redémarrage. Il est donc indispensable que le fichier contenant la clé ne soit lisible que par le propriétaire des process apache (qui est déclaré dans le fichier de configuration d''apache ***httpd.conf***).

- Placez vous dans un répertoire vide pour éviter tout désagréments et entrez:

```
/usr/bin/openssl genrsa 1024 > nom.votre.site.key
```

où évidemment nom.votre.site est le nom "homologué" de votre site web.
Si vous souhaitez créer une clé cryptée, entrez:

```
/usr/local/bin/openssl genrsa -des3 1024 > nom.votre.site.key
```

- Mais dans ce cas, chaque lancement d'apache vous demandera d'entrer manuellement le mot de passe de la clé.

```
[root@compaq ssl]# /usr/bin/openssl genrsa -des3 1024 > compaq.cybergate.local.k
ey
Generating RSA private key, 1024 bit long modulus
.....++++++
.....................++++++
e is 65537 (0x10001)
Enter PEM pass phrase:
Verifying password - Enter PEM pass phrase:
[root@compaq ssl]# _
```

Cela peut être gênant si vous prévoyer des redémarrages automatisés d'Apache....
Il y a un moyen d'automatiser la fourniture du mot de passe à Apache au démarrage avec l'option:

```
SSLPassPhraseDialog exec:votre_programme_de_fourniture_de_mot_de_passe
```

- Mais c'est à vous d'écrire le programme qui fournit le mot de passe, faites attention à ce que vous faites...

- entrez ensuite:

```
/usr/local/bin/openssl req –new \
-key nom.votre.site.key \
-out nom.votre.site.csr
```

- qui va se servir du fichier de configuration par défaut de openssl
- Vous devez répondre à une série de questions, entrez un **« . »** pour laisser un champ vierge, voici ce que j'y ai entré:

```
Country Name (2 letter code): FR
State or Province Name :
Locality Name: PARIS
Organization name: Cybergate Paradize
Organization Unit Name:
Common name: www.cybergate-paradize.fr
Email Adress: david@cybergate-paradize.fr
A chalenge password:
An optional company name:
```

- Le CSR est maintenant créé, on peut valider soi même pour générer un certificat de sécurité en entrant:

```
/usr/local/bin/openssl req -x509 -days 365 \
-key cybergate-paradize.fr.key \
-in cybergate-paradize.fr.csr \
-out cybergate-paradize.fr.crt
```

- Le -days donne la durée de validité du certificat, 365 dans ce cas
  Il faut déplacer les fichiers crées dans le répertoire d'installation d'apache
- Des répertoires spécifiques ont été créés à l'installation.

```
cybergate-paradize.fr.crt dans /usr/local/apache/conf/ssl.crt
cybergate-paradize.fr.csr dans /usr/local/apache/conf/ssl.csr
cybergate-paradize.fr.key dans /usr/local/apache/conf/ssl.key
```

- Vérifiez bien que ces fichiers ne sont lisibles que par le propriétaire des process apache (surtout si vous utilisez une clé non cryptée)
- L'installation est à présent terminée.Il ne reste plus qu'à configurer le fichier /usr/local/apache/conf/httpd.conf

# ☯ MODSSL

Lieu : ftp://ftp.modssl.org/source/ mod_ssl-2.8.12-1.3.27.tar.gz
Manuel : http://www.modssl.org/docs/

## I. INTRODUCTION

Le mod ***SSL*** est en fait un patch pour ***Apache***.
Pour l'appliquer, il suffit d'avoir les sources d'apache et de commencer l'installation dans le répertoire contenant les sources de ***mod_ssl***

## II. INSTALLATION

```
tar xvzf mod_ssl-2.8.12-1.3.27.tar.gz
cd mod_ssl-2.8.12-1.3.27
./configure \
--with-apache/usr/local/src/apache-1.3.27 \
--with-ssl=/usr/local/src/openssl-0.9.7e \
--prefix=/usr/local/apache \
--enable-module=ssl \
--enable-module=most \
--enable-shared=max \
--enable-rule=EAPI
```

```
cd /usr/local/src/apache-1.3.27
make
```

## III. CREATION DES CLES

| | |
|---|---|
| **make certificate** | Permet de générer les différents fichiers servant de clés. |

- Répondre aux questions

```
RSA [R] : R
Country Name : FR
Locality Name: Paris
Common name: www.cybergate-paradize.fr
Email: root@cybergate-paradize.fr
Certificate validity: 10000
Certificate version [3]: 3
Encrypt the private key now?: Y          Encrypter la clé privée
PEM pass phrase:                         mettre un mot de passé pour l'activation
```

- Une fois cette procédure terminée, vous trouverez différents répertoires dans ***/usr/local/apache/conf***
  *ssl.crl*
  *ssl.crt*
  *ssl.csr*
  *sss.key*
  *ssl.prm*

ces répertoires contiennent les fichiers générés par ***make certificate***

```
make install
```

- Ensuite il faut Installer ***PHP***

SI vous regarder attentivement ce qui se passe lors de la compilation des différents fichiers, vous remarquerez la présence d'une option qui nous laisse penser que tout se passe bien ***(-DMOD_SSL=208110 et -DEAPI).***

```
make install
cp php.ini-dist /usr/local/lib/php.ini
```

Puis modifiez php.ini comme suit afin d'éviter certains problèmes de passage de paramètres en URL : /•

***Register_global = Off --> Register_globals = On*** **lignes 309**

- 

## IV. CONFIGURATION D'APACHE /PHP EN MODE SSL

- Pour que nous le serveur apache puisse convenablement interpréter les fichiers ***.php*** il reste plus qu'a rajouter quelques lignes de configuration dans **/usr/local/apache/conf/httpd.conf**

Vous remarquerez au passage que ce fichier à sérieusement grossit ces derniers temps et qu'il contient bon nombre de renseignement sur ***SSL***.
Voici lignes à rajouter :

```
<IfModule mod_dir.c>
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
</IfModule>
```

```
IfDefine SSL>
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
</IfDefine>
```

Ces deux blocs permettent de définir les types ***MIME*** dans deux cas de figure particuliers. Le premier permet de gérer les fichiers de type ***.php*** dans le cas où **apache** est lancé en mode standard.
Le second permet de gérer les fichiers de type ***.php*** dans le cas où apache est lancé avec le mode ***SSL***.
Néanmoins, une dernière petite chose reste à modifier.
Si vous regarder attentivement le fichier de configuration, vous verrez que les modules gérant le **php** ne sont chargés qu'à partir du moment où nous démarrons apache avec le mode **SSL**.
Pour rectifier cela modifier les blocs de code suivants :

```
IfDefine SSL>
LoadModule ssl_module     libexec/libssi.so
LoadModule php4_module    libexec/libphp4.so
</IfDefine>
```

en

```
IfDefine SSL>
LoadModule ssl_module     libexec/libssi.so                    ligne 224
</IfDefine>
LoadModule php4_module        libexec/libphp4.so
```

Et

```
<IfDefine SSL>
AddModule mod_ssl.c
AddModule mod_php4.c
</IfDefine>
```

en

```
<IfDefine SSL>
AddModule mod_ssl.c                                    ligne 250
</IfDefine>
AddMod.ule mod_php4.c
```

- Voilà, désormais les fichiers de type ***php*** seront gérés dans toutes les configurations d'apache.

Une dernière petite chose à faire, mais cette fois-ci plus pour le confort, ajouter comme pages de démarrage, les fichiers index, php

```
<IfModule mod_dir.c>
DirectoryIndex  index.php index.html                   ligne 422
</IfModule>
```

Il ne vous reste plus qu'à démarrer apache en mode normal et en mode ssl pour voir si tout s'est bien passé.

```
usr/localapache/bin
./apachectl start
```

Pour tester le tout, il ne vous reste plus qu'à créer un fichier php et à le tester dans tous les modes d'apache. ,
Remplacer ***public/html*** par le repertoire voulu ***/home/http/***

- http://127.0.0.1/info.php

Par exemple ***Info.php*** contenant le code suivant :

```
<?
phpinfo() ;
?>
```

- Lancer

```
 usr/localapache/bin
./apachectl startssl
```

- https://127.0.0.1/info. php

## V. CREER UN BLOC VIRTUALHOST

Rajouter à la ligne 1133 à 1140 dans ***httd.conf***
Copier obligatoirement les 6 cl és comportant une étoile

```
<VirtualHost>
SSLEngine On                                                    ***
SSLCipherSuite                                                  ***
SSLCertificateFileconf/ssl.crt/webmail. nom.domaine.net.crt     ***
SSLCertificateKeyFileconf/ssl.key/webmail.nom.domaine.net.key   ***

DocumentRoot        /webnew/webmail
ServerName          nom.domaine.net
CustomLog           /var/log/httpd/mailJog common
ErrorLog            /var/log/httpd/mail-errorJog
SetEnvIf            User-Agent ".*MSIE.*"\                      ***
nokeepalive         ssl-unclean-shutdown \
downgrade-1.0 force-response-1.0

        <Files ~ "\.(cgi|shtml|phtml|php3?|php|inc)$">
        SSLOptions +StdEnvVars
        </Files>
        <Directory"/usr/local/apache/cgi-bin">
        SSLOptions +StdEnvVars
        </Directory>
</VirtualHost>
```

Dans votre explorateur: https://phpmyadmin.nom_de_domaine:443

***SSLEngine on*** L'activation de ***ssl*** se fait par l'ordre
***SSLCipherSuite*** la localisation des fichiers de certificat,
***DocumentRoot - ServerName - CustomLog – ErrorLog*** les ordres "classiques" des serveurs virtuels
***SetEnvIf ,nokeepalive***
Des options de configuration automatiquement par la compilation d'apache dans l'hôte virtuel "par défaut"

## VI. LE FICHIER .HTACCESS

- dans /apache/htdocs/privé, créer un fichier ***.htaccess***

- créer un fichier de groupe
- créer fichier d'utilidsateur apache différent des utilisateurs systèmes

```
mkdir /usr/local/apache/auth
/usr/local/apache/bin/.htpasswd
htpasswd –c /usr/local/apache/auth/.htpasswd user1
vi .htgroupes            groupe: user1  user2 user3  …
vi .htaccess
```

```
AuthUserFile   /usr/local/apache/auth/.htpasswd
AuthGroupFile/usr/local/apache/auth/.htgroupe
AuthName       AccesRestreint
AuthType       Basic
        <limit Get>
        order deny,allow
        deny from all                          Interdit tout le monde
        require groupe mon_groupe              Autorise mon groupe
        satisfy all                            Satisfiera à toutes les requêtes
</limit>
```

- Faire un chmod sur les fichier suivant :

```
chmod 755 .htaccess
chmod 755 .htpasswd
chmod 755 .htgroupe
phpmyadmin
```

## VII. PROBLEME

Si la boite ne s'affiche pas, modifier à la ligne 385 de ***httpd.conf***

```
allow override none
```

en

```
allow override all
```

# ☯ SGBD

## I. INTRODUCTION AUX BASES DE DONNEES

La fiche : Contient une identification ***ID,*,*** Obligatoire, est incrémenté automatiquement
***Nom,Prénom,Adresse,Tel,Email,CP,Ville,Fax***

Les fiches sont contenues dans des ***tables***, qui eux-mêmes contiennent des ***enregistrement*** qui eux-mêmes contiennent de ***champs***.

*Exemple :* Table employés

| ***Champs*** | ***Type*** |
|---|---|
| *ID* | *INT (ebtier) Auto (incrément)* |
| *Nom* | *Text* |
| *Prénom* | *Text* |
| *CP* | *Text* |
| *Ville* | *Text* |
| *Tel* | *Text* |
| *Mail* | *Text* |

## II. LANGAGE SQL

***Structure Query Langage***

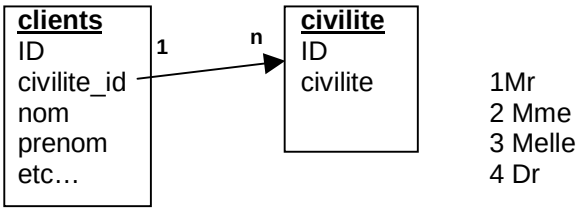
Manuel : Teach Yourself in 10 mn SQL 7.50€ environ

### A. Type de recherche

- Sélectionner la table client
  ***SELECT * FROM Clients ;***
  - Sélectionner les clients qui se prénomment Pierre
    ***SELECT * FROM Clients WHERE clients.prenom #LIKE 'pierre';***

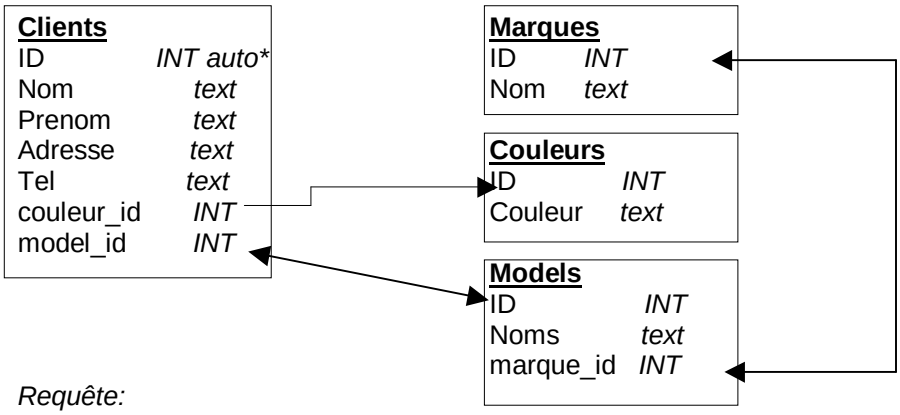
    Toujours spécifier la ***table*** avant le ***champs***
    - Et qui habitent Paris
      ***AND clients.ville LIKE 'Paris';***

### B. Requetes sur plusieurs tables




Exemple : ***SELECT * FROM clients,civilites WHERE employés.civilite_id = civilites.ID ;***

## III. EXEMPLE DE BASE DE DONNEES




*Requête:*
*SELECT client.nom, client.prenom,client.adresse,client.tel,client.model_id*
*select * FROM clients,couleurs,models,marque WHERE clients.model_id = models_ID*
*AND models.marque_id = marques.ID AND clients.couleurs_id = couleurs.ID*

# ☯ PROBLEMES RENCONTRES

- Si ***GD*** est mal installé, modifier dans le fichier ***Makefile*** de ***GD***
  ***INCLUDE_DIRS*** ***/usr/local/lib***
  ***LIB_DIRS*** ***/usr/local/lib***
  ***INSTALL_DIRS*** ***/usr/local/lib***

- Lors de l'installation de PHP, il est possible qu'il soit utile d'installer le package ***libtools***

- Demande d'enregistrement du fichier PHP
  vérifier le ***AddType*** dans ***httpd.conf***
  PHP est mal installé

- Mauvaise interprétation de PHP
  vérifier le ***AddType***

# ☯ CONCLUSION

Ce guide, bien qu'il ne soit pas exhaustif, a pour vocation de présenter comment installer une plate-forme de développement de sites dynamiques sur une plate-forme ***Linux***. et ***OpenBSD avec mode SSL***

- ***Installation des librairies***
- ***Mysql***
- ***Apache***
- ***PHP***
- ***PHPMyAdmin***
- ***Mod_SSL***
- ***Open_SSL***

Si vous rencontrez une quelconque difficulté ou si vous avez des questions ou des remarques, n'hésitez pas à poster un message dans le Forum ou a m'envoyer un message à l'adresse dparize@yahoo.fr.

Sites utiles:
WWW.LINUX-SOTTISES.NET
www.toutestfacile.com

# ☯ INDEX